MICROCOMPUTADOR CURSO PRÁTICO

TIRE O MÁXIMO DO MICRO EM 30 SEMANAS

PUBLICAÇÃO SEMANAL ILUSTRADA

11

Cr$ 8.200

A estratégia dos computadores

1+2=20? • Teste sua memória

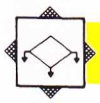
rioGráfica

MANAUS, ALTAMIRA, MACAPÁ, JI.PARANÁ, PORTO VELHO, RIO BRANCO, SANTAREM, BOA VISTA, SINOP, ALTA FLORESTA. Cr$ 10.660

MICROCOMPUTADOR CURSO PRÁTICO

APLICAÇÕES

Xadrez nos chips 217



LINGUAGENS

1 + 2 = 20? 220

RAIO X

Wafadrive 225

SOFTWARE

Controle total 228

PROJETOS DE PROGRAMAS

Cores e números 230

CIÊNCIA DA COMPUTAÇÃO

Mudanças de códigos 232

TÉCNICAS DE PROGRAMAÇÃO

Ajuda em linha 234

PERFIL

Objetivo brasileiro 236

TESTE

Depois de ler com atenção todo o fascículo, responda a estas perguntas. Veja a solução no próximo número, junto com um novo teste.

1) Quais as duas características mais notáveis da fita utilizada pelo floppy-tape Wafadrive?
2) Qual a alternativa para os softwares integrados que apresenta os mesmos resultados ocupando menos memória?
3) Quais os dois comandos em PASCAL que controlam loops condicionais?
4) Qual o comando em PASCAL que transformaria o número 56.999 em 56?

RESPOSTAS DO TESTE ANTERIOR

1) A impressora do tipo margarida produz caracteres mais legíveis e de diferentes tipos, enquanto a matricial é bem mais rápida.
2) As quatro leis básicas da álgebra booleana são: a associativa, a comutativa, a distributiva e a de De Morgan.
3) O acoplador acústico transmite dados por meio do auscultador de um telefone, enquanto o modem liga-se diretamente à linha telefônica.
4) Para trabalhar com decisões que envolvem opções múltiplas, o PASCAL dispõe da instrução CASE-OF.

PLANO DA OBRA

MICROCOMPUTADOR - CURSO PRÁTICO é uma coleção de 30 fascículos de periodicidade semanal. Os 29 primeiros terão, cada um, vinte páginas internas (miolo) e quatro capas, além de uma introdução de dezesseis páginas acompanhando o fascículo 1. O fascículo 30 incluirá o índice geral da obra. Os miolos, encadernados, constituem dois volumes de um curso prático de microcomputação, com seções indicadas por tarjas de diferentes cores.
As duas primeiras capas são descartáveis; as 3.as e 4.as trazem programas de jogos. Encadernadas, formam um volume de cem páginas.

COMO ENCADERNAR

As capas duras, incluindo as guardas, estarão à venda simultaneamente com os fascículos 10 (**Volume 1**) e 30 (**Volume 2** e **Jogos**).
Volume 1 - Deve ser encadernado com os elementos dispostos nesta ordem: guardas; frontispício e introdução (publicados com o fascículo 1); miolos dos fascículos de 1 a 15; e guardas.
Volume 2 - Guardas; frontispício (que virá com o fascículo 30); miolos dos fascículos de 16 a 30; e guardas.
Jogos - Guardas; frontispício e sumário (que constituem as 3.as e 4.as capas do fascículo 30); 3.as e 4.as capas dos fascículos de 1 a 29; e guardas.

Nota: Muitos dos equipamentos que aparecem na seção RAIO X não são normalmente comercializados no Brasil. Sua divulgação é feita a título de informação ao leitor.

rioGráfica **Diretoria Executiva:** Oscar Neves; Filipe Zander; Nilo Sérgio de Almeida; Danilo Esteves Costa; **GRUPO EDUCAÇÃO E CULTURA - Planejamento e Comercialização - Gerente de Planejamento:** João Azeredo Coutinho; **Gerente de Marketing:** Jaime Rodrigues; **Gerente de Produto:** José Mauro Porto; **Gerente do P.C.P.:** Aylton Menezes; **Diretor de Vendas:** Marcos March; **Gerente de Vendas:** Rubens Barbosa; **Coordenadores de Promoção:** Paulo Cesar M. Seixas; Edgar Mello M. Neto; **Gerente de Circulação:** Norberto Martin; **Supervisor de Operações:** Abel Pereira Pinto; **Editorial - Diretor:** João J. Noro; **Editor-chefe:** Mauricio Rittner; **Administração Editorial:** Heitor de Souza Paixão; **Editor:** Jesse Navarro; **Revisores:** Isis Augusta Loyolla; Cacilda Guerra; Luiz Vicente Vieira Filho; Marcos Domingos Agathão; Maria da Graça Mendonça Couto; **Diretor de Arte:** Adalberto Cornavaca; **Chefe de Arte:** Bonifácio Duardes Miranda; **Chefe de Estúdio:** José Yuji Kuribayashi; **Assistente de Arte:** Miguel Luis Escámez Simón; **Colaboradores - Editores-assistentes:** Claudio Marcondes; Isa Mara Lando; Luiz Carlos Pizarro Marin; Nadya Morelo Reis; **Preparação de Texto:** Luiz Carlos Cardoso; **Tradução:** Fernando Dorfman Knijnik; Lúcia Maria Leal Gonçalves; Maria Clara Cescato; Maria Ester Menezes Martinho; Newton Roberval Eichemberg; Patrícia de Paiva e Castro; Regina Amarante; **Pesquisa:** Stella Maria Quentel; **Arte:** Edemilson R. da Silva; **Assessoria Técnica:** Geraldo Coen; Gustavo José Ferreira Grubba; Lauro de Lauro; Pierluigi Piazzi; Sérgio Jorge Batarce; Imarés Microcomputadores. As fotos não creditadas pertencem à obra original. Seguem-se os créditos das demais: Fernando Scavone; Scopus; Editora Abril. © APSIF Copenhagen, 1984; © Orbis Publishing Ltd., 1984; © Editora Rio Gráfica Ltda., 1985, para a língua portuguesa. **Composição:** Editora Rio Gráfica Ltda. e AM Produções Gráficas Ltda. **Impressão:** JBIG.

NÚMEROS ATRASADOS - Você poderá comprar os exemplares que faltam em sua coleção, pelo preço do último fascículo posto à venda, em sua banca de jornal preferida. Caso não consiga obtê-los, faça sua solicitação por carta endereçada diretamente à Editora Rio Gráfica Ltda., rua Itapiru, 1209, Rio Comprido, Rio de Janeiro, CEP 20251. O atendimento será feito por via postal. Nas cidades do Rio de Janeiro e de São Paulo, as compras poderão ser efetuadas pessoalmente nos seguintes endereços: Rio de Janeiro - Rua Itapiru, 1209, Rio Comprido; São Paulo - Rua Frei Caneca, 1152, Consolação.

Distribuidor exclusivo para todo o Brasil: Fernando Chinaglia - Rua Teodoro da Silva, 907, Rio de Janeiro. Distribuidor para Portugal: Electroliber Ltda. - Rua Prof. Reinaldo dos Santos, 1488, Lisboa.

XADREZ NOS CHIPS

Batalha no tabuleiro
Mestre internacional de xadrez, David Levy abandonou as competições contra jogadores humanos em 1978. Em 1968, apostou enorme quantia em sua habilidade, desafiando qualquer programa de xadrez para computador a vencê-lo nos dez anos seguintes. O período abrangido pela aposta ampliou-se, mas o enxadrista permanece invicto. Considerado uma autoridade em xadrez para computador, Levy dirige a Intelligent Software, empresa que pesquisa técnicas de programação interativas. Muitos dos computadores dedicados ao xadrez e programas para xadrez em micro baseiam-se nessas técnicas. Levy acredita que, em termos de desempenho no jogo de xadrez, os micros aproximam-se agora dos computadores de grande porte.

Esta matéria examina o raciocínio e algumas das idéias que possibilitaram o desenvolvimento de programas para um dos mais antigos e conceituados jogos de estratégia em tabuleiro.

Poucos jogos seduziram a imaginação tanto quanto o xadrez: milhões de pessoas em todo o mundo o jogam há milhares de anos e suas regras atuais são quase as mesmas usadas desde o século XVII. Há os que dedicam a vida ao estudo desse jogo de estratégia, encontrando satisfação em sua exigência de agilidade mental.

O xadrez originou uma série de variações que buscam introduzir níveis maiores de complexidade. Por exemplo: o xadrez tridimensional envolve vários tabuleiros suspensos no espaço e exige concentração muito maior. Outra variante é o xadrez com três pessoas, jogado num tabuleiro em forma de Y. Nas diagonais, onde as três "alas" se intersectam, há regras especiais para o movimento das peças. A teoria por trás dessa versão é de que dois jogadores se unam contra um terceiro e depois lutem um contra o outro pela vitória. Mas nenhuma dessas variações conseguiu substituir o confronto entre duas pessoas jogando no tabuleiro com 64 quadrados.

Uma das razões disso está no número quase infinito de variações possibilitadas pelo próprio jogo. Em 1949, o matemático Claude Shannon calculou que há 10^{120} jogos possíveis com quarenta movimentos cada um. Isso significa que uma pessoa que jogasse xadrez 24 horas por dia, sete dias por semana, despendendo uma hora para cada jogo (o que não é tempo demais para quarenta movimentos), levaria mais de 10^{17} anos para fazer os jogos possíveis.

Dada a complexidade, não é de surpreender que a programação de computadores para jogar xadrez tenha exigido muito tempo e esforço. Os programas de xadrez já são executados em computadores de grande porte há muitos anos e existem agora inúmeras versões para micros.

O desenvolvimento de programas de alta qualidade para micros vincula-se às inovações no hardware. As áreas problemáticas sempre foram a ausência de capacidade suficiente de memória e a velocidade relativamente lenta do processamento. Mas os avanços da tecnologia nos últimos anos fizeram com que a qualidade de tais programas dependa, agora, do software.

Como os computadores são calculadoras que operam em alta velocidade, o xadrez para computador é projetado com base em cálculos numéricos, usados para avaliar os dois elementos essenciais do jogo: o material e a mobilidade.

Estratégia em software

Para jogar xadrez contra o computador é preciso, em geral, conhecer as regras do jogo. Isso porque a maioria dos softwares disponíveis, tanto no mercado nacional como no estrangeiro, classifica os jogos de xadrez por níveis.

Os do nível 1 ao nível 8 são elaborados para jogadores de conhecimentos técnicos medianos e compõem a totalidade dos programas de xadrez oferecidos no mercado nacional. Existem jogos de níveis superiores, importados, que incluem lances de campeonatos internacionais, e alguns até acima do nível de disputa entre campeões.

Entre os programas nacionais, destacam-se dois jogos de xadrez Microsoft, desenvolvidos para micros TK 83/85 e TK 2000.

Para utilizar um jogo de xadrez no TK 2000 Color, é necessário um gravador mono e um cabo ligando o monitor, pela saída EAR. Depois de passar a fita com o programa para a memória do micro, surgirão na tela o tabuleiro e as instruções para iniciar o jogo.

Conhecendo-se as regras, o jogo da Microsoft é muito simples. De início, o computador apresenta na tela o tabuleiro com as peças colocadas. (Alguns softwares estrangeiros, como o Colossus, não apresentam esse recurso.) As peças do jogador encontram-se na parte inferior do tabuleiro, mas essa

O material de um jogo refere-se à quantidade e ao valor das peças no tabuleiro. O programa de xadrez atribui a cada peça um valor numérico. O rei pode receber um valor infinito ou um valor arbitrário alto, como 10.000 (isso porque a perda iminente dessa peça encerra o jogo); a rainha recebe o valor nove; a torre vale cinco; os bispos e os cavalos, três; e os peões, um.

Para decidir se vale a pena sacrificar uma peça a fim de capturar uma das peças do oponente, o programa compara seus valores. A maioria dos programas de xadrez para computador dá grande importância aos valores relativos e raramente trocará peças se isso resultar em desvantagem material, a menos que haja um avanço acentuado de posição.

A mobilidade é fator muito importante no xadrez: qualquer peça terá pouco valor se seu movimento estiver limitado. Inversamente, seu valor aumentará se ela puder ser posicionada de forma que assegure influência em várias áreas do tabuleiro ao mesmo tempo. O programa de xadrez precisa, desse modo, avaliar a mobilidade, assim como os aspectos materiais. Além disso, o programa deve planejar com antecedência, determinando a melhor seqüência de movimentos a partir de qualquer posição.

É aí que os programas de xadrez se mostram notáveis, usando a velocidade de processamento do computador para examinar um grande número de movimentos possíveis em muito menos tempo do que despenderia uma pessoa.

A maioria dos programas de xadrez usa uma técnica de "força bruta", pesquisando o maior número de movimentos possível no tempo permitido. A duração de cada movimento é determinada pela escolha do nível de jogo, cada um dando um espaço de tempo diferente durante o qual o computador deve fazer um movimento.

Esses períodos variam de alguns segundos até várias horas e, quanto mais tempo tiver para a pesquisa, tanto mais probabilidade o computador terá de encontrar a melhor linha de ataque para a posição atual.

A cada movimento, o computador verifica se o rei está ou não em xeque e a seguir examina que probabilidade as peças têm de ser ganhas ou perdidas, se há casas importantes que podem ser ocupadas e outras questões semelhantes. Quanto mais critérios o programa examinar, melhor será o resultado. A questão final é a de descobrir se o rei do oponente pode ser forçado a uma posição de xeque-mate.

Nos jogos entre computadores e seres humanos, a máquina tem acentuada vantagem em termos de velocidade e amplitude de pesquisa — mesmo assim, um excelente jogador humano sempre vence um excelente programa de xadrez para computador, devido a sua capacidade de visualizar e criar novas aberturas e posições. Os

posição pode ser invertida teclando-se [X] seguido de [Return]. O jogador escolhe o nível de jogo digitando [Ql] = [1] ou qualquer algarismo até 8. Se você digitar [N], o computador mostrará as coordenadas dos quadrados do tabuleiro, utilizando, na horizontal, as letras de A a H e, na vertical, os números de 1 a 8.

Para movimentar as peças, é necessário digitar as coordenadas que indicam sua posição inicial seguida das coordenadas da posição final; depois, basta digitar [Return] para confirmar a jogada. Por exemplo, digitando-se [B5-B7], a peça que está no quadrado B5 move-se para o B7. Erros de digitação podem ser corrigidos por meio da tecla [<]. Se algum dos movimentos for ilegal, o computador acusará com um ponto de interrogação e um bip, exigindo nova jogada. O jogador pode relegar ao computador a tarefa de jogar por si digitando [P] seguido de [Return]. O programa permite ainda o roque, que é executado sem consulta quanto à legalidade. Para realizar um roque pequeno, digita-se [0-0]; para um roque grande, [0-0-0], sempre seguidos de [Return].

É possível ainda analisar cada situação de tabuleiro, o que se faz acrescentando um asterisco ao final de cada movimento. Nesse caso, o computador não testa a legitimidade do movimento.

Características	Grand Master 64	Colossus	Cyrus IS	Sargon III
Jogo invisível	NÃO	SIM	NÃO	NÃO
Lista os movimentos	NÃO	SIM	NÃO	NÃO
Reapresenta um jogo	NÃO	SIM	SIM	SIM
Mostra a listagem	NÃO	SIM	NÃO	SIM
Imprime a listagem	NÃO	NÃO	SIM	SIM
Retoma posições	SIM	SIM	SIM	SIM
Treinamento	SIM	NÃO	NÃO	SIM
Promove peão	NÃO	SIM	SIM	SIM
Imita o jogo humano	NÃO	SIM	SIM	SIM
Muda de lado	SIM	SIM	SIM	SIM
Analisa problemas	NÃO	SIM	SIM	SIM
Mostra pesquisa	LANCE ÚNICO	SIM	NÃO	SIM
Inverte o tabuleiro	SIM	SIM	SIM	SIM
Oferece empate	NÃO	NÃO	NÃO	SIM
Imprime o tabuleiro	NÃO	NÃO	SIM	SIM
Grava o jogo	NÃO	SIM	SIM	SIM
Relógio de tempo real	SIM	SIM	NÃO	SIM

Xadrez internacional

Um bom programa de xadrez inclui roque, promoção de peão, captura **en passant**, e compreende situações de empate e de xeque perpétuo. O quadro acima mostra algumas das possibilidades de quatro jogos de xadrez para computador disponíveis no mercado internacional.

computadores jogam um excelente xadrez tático, mas, mesmo entre os mestres humanos do xadrez, o jogador que se posiciona bem costuma vencer o bom jogador tático.

Os programadores de xadrez para computador concentraram-se na tática porque, para o computador, o jogo tático envolve apenas lidar com números. Se um opositor humano faz um movimento não convencional, o computador quase sempre falha na escolha da melhor resposta. Por isso, muitos programas de xadrez têm dificuldade em lidar com posições estáticas, em que nenhum dos movimentos possíveis oferece vantagem tática específica. Nessas situações, o programa muitas vezes apenas muda as peças de lugar, em vez de aproveitar a oportunidade de planejar sua estratégia.

Um estilo de programação recentemente desenvolvido recorre à pesquisa seletiva. Usando essa técnica, o computador imita o jogador humano: examina com maior atenção um número menor de movimentos possíveis. O programa Mephisto III utiliza a técnica de pesquisa seletiva para os dois primeiros movimentos; depois, restringe a pesquisa e examina um conjunto menor de movimentos em detalhe. O Mephisto III também tenta distinguir entre posições táticas e estáticas — técnica que, acredita-se, tornará os computadores do futuro um verdadeiro desafio aos jogadores humanos.

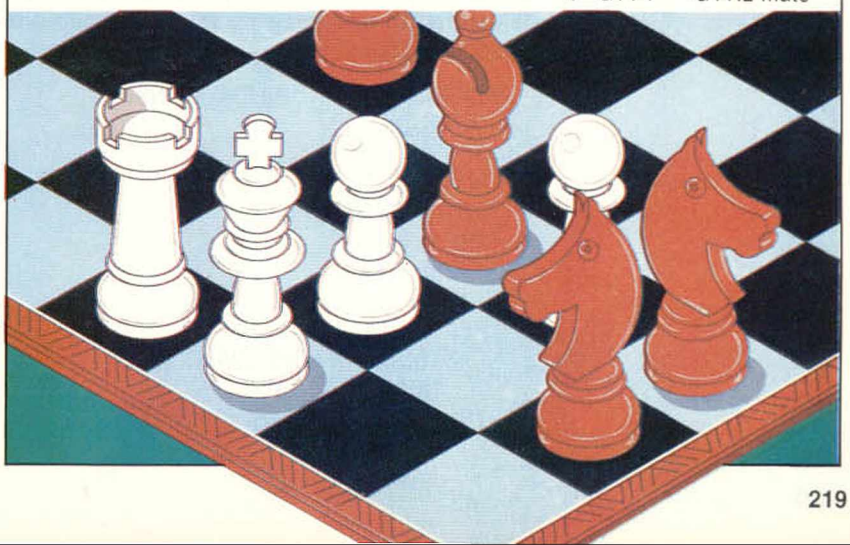
Coração x cabeça

A capacidade de examinar todas as posições com até nove movimentos de antecipação quase garante aos programas de xadrez uma superioridade tática com relação aos seres humanos. A técnica dos mestres enxadristas consiste em selecionar alguns movimentos decisivos nos quais concentram enorme capacidade de análise, com até trinta movimentos de antecipação.

A ilustração mostra um movimento decisivo do jogo de Moritz (pretas) contra Emmerich (brancas), em 1922. As pretas podem dar xeque-mate sacrificando a rainha e fazendo três elegantes movimentos com o cavalo; a maioria dos jogadores preferiria, sem hesitar, essa seqüência. O próprio Moritz não a percebeu e lamentou amargamente seu descuido. Os programas chegariam ao mate, sem, contudo, sugerir os lances com os cavalos.

A seqüência dos movimentos:

1		H5-H2 xeque
2	G1-H2	E5-G4 xeque
3	H2-G1	F4-H3 xeque
4	G1-F1	G4-H2 mate



1+2=20?

A aplicação dos vários tipos de dados a cálculos aritméticos e o controle de loops do programa por estruturas iterativas são os recursos da linguagem PASCAL que examinaremos em pormenores neste artigo.

A afirmação de que um mais dois é igual a três pressupõe que estamos lidando com números naturais puros. Mas, quando se utiliza uma nota de 1.000 cruzeiros e duas de 500, o resultado da operação, sob outro ângulo, pode ser um maço de cigarros. Esse tipo de classificação de dados é fundamental na descrição de qualquer problema e o PASCAL dispõe de vários recursos para organizá-los e descrevê-los de modo claro e lógico.

A criação de novas categorias de dados torna possível a solução do problema, facilitando o projeto dos algoritmos com base no processamento adequado a cada categoria. Se, por falta de atenção, tentarmos fazer algo sem sentido — por exemplo, acessar um valor booleano por meio do teclado —, o compilador PASCAL interceptará de imediato esse erro lógico. Muito trabalho é poupado evitando a execução de qualquer instrução antes que todos os erros tenham sido eliminados do programa-fonte.

Para maior proveito das descrições de dados "fortemente tipificadas", o usuário do PASCAL pode impor algumas restrições adicionais às variáveis. Em resumo, as regras para manipulação dos tipos simples de variáveis são: nenhum dos escalares é "compatível" com variáveis de qualquer outro tipo escalar, embora os dois tipos numéricos possuam muitas características comuns; e, assim como os números reais podem ser arredondados para inteiros, é possível a atribuição de valores do tipo integer a reais, mas não o inverso. Eis alguns exemplos:

```
PROGRAM COMPATIBILIDADE (INPUT,
  OUTPUT);
VAR
  INTA,
  INTB :INTEGER;
  XREAL,
  YREAL :REAL;
BEGIN
  READ (INTA, XREAL); {LE UM
    INTEGER, DEPOIS QUALQUER
      NUMERO LEGAL}
  YREAL:=INTA; {REAL:=INTEGER
    E CORRETO}
  INTB:=XREAL; {** ERRO : ILEGAL **}
  {.. ETC.}
```

As operações aritméticas são definidas apenas nos tipos numéricos. Tanto para os valores integer como para os reais podem-se usar os quatro operadores simbólicos habituais:

+ (adição);
− (subtração);
* (multiplicação);
/ (divisão com vírgula flutuante).

Nesse contexto, tornam-se operadores "diádicos" ou binários, pois sempre exigem dois "operandos" de qualquer um dos tipos numéricos. Toda vez que um dos operandos for real, o resultado da expressão também o será, de modo que: 2+2,0 serão 4,0 (e não 4). No caso do símbolo de divisão, a expressão sempre resultará em um número real, mesmo quando os dois operandos forem integers: 3/5 será 0,6, 8/4 serão 2,0.

Na divisão de valores inteiros, muitas vezes a resposta "real" não faz sentido. Por exemplo, doze barras de chocolate divididas entre dez pessoas resultarão em uma para cada pessoa, restando duas barras. O resultado integer e o resto podem ser obtidos com os dois operadores de divisão integer DIV e MOD (ambos palavras reservadas do PASCAL): 15 DIV 5=3 e 15 MOD 5=0; 31 DIV 7=4 e 31 MOD 7=3.

Preste atenção para a possibilidade de os dados serem negativos, pois não se pode realizar uma divisão integer quando o denominador assume valores negativos. Na execução do programa, qualquer tentativa de dividir por zero resultará numa mensagem de erro — pelo menos enquanto não se inventar um modo de avaliar o infinito.

Em toda expressão, as operações de multiplicação e divisão são efetuadas antes das de adição e subtração. Emprega-se a notação com parênteses para inverter essa "prioridade" lógica. Por exemplo, (8+4) DIV 2=6, mas 8+4 DIV 2=10.

Funções de transferência

Embora não possamos fazer uma atribuição direta de um valor real a uma função integer, o PASCAL possui "funções de transferência" muito úteis para isso — os identificadores TRUNC e ROUND. A função TRUNC elimina (trunca) a parte fracionária de um número real, qualquer que seja ela; por exemplo, TRUNC (3,999) = 3 e TRUNC (−123,456) = −123.

A função ROUND executa um arredondamento "inteligente": para mais se a parte fracionária for maior que 0,5, e para menos em caso

contrário. Então, ROUND (1.234,6) = 1.235 e ROUND (−0,49237) = 0.

Ocorrerá erro se o argumento real para qualquer dessas funções resultar num valor inteiro fora da faixa do tipo integer (−MAXINT a MAXINT); assim, sempre verifique antes utilizando a instrução IF. Do mesmo modo, o argumento precisa ser real e não integer, pois já terá sido truncado e arredondado.

O PASCAL possui a função ODD (ímpar), que fornece um resultado booleano — falso se seu argumento integer for um número par, e verdadeiro se for ímpar:

```
IF ODD (N)
  THEN
    WRITELN ('E IMPAR!')
  ELSE
    WRITELN ('ISSO TORNA PAR!')
```

ou:

```
IF ODD (N) THEN
  IF N MOD 2=0 THEN
    WRITELN ('CONSERTE O COMPILADOR!')
```

A tabela da página seguinte mostra todas as funções aritméticas do PASCAL. Elas podem assumir qualquer argumento numérico escalar, real ou integer. As duas primeiras, ABS e SQR, fornecerão um valor de tipo compatível com seu argumento. Assim, ABS (−19,372) será 19,372 e ABS (255) será 255.

As outras funções, porém, sempre fornecerão um resultado real, de modo que SQRT (16) será 4,0 e não 4. Isso porque os algoritmos usados no cálculo das funções baseiam-se na soma de uma série de termos, todos fracionários. Nas referências aos resultados reais das funções aritméticas, deve-se dizer "será o mesmo que" em vez de "igual a", pois não é seguro comparar valores reais pela igualdade exata. O menor erro nos cálculos significará que dois valores reais nominalmente "iguais" podem de fato diferir por, digamos, 1,0E−27. Erro insignificante, mas o computador não sabe disso. Em vez de usar IF X=Y THEN.., podemos usar a função predefinida do PASCAL ABS:

```
IF ABS (X-Y)<QUASE ZERO THEN {ETC.}
```

Quando a diferença entre X e Y for muito pequena, recomendamos o seguinte teste. A aproximação considerada desprezível pode ser especificada com proveito mediante uma definição CONST no início do programa. Os logaritmos são fornecidos na base natural (e), não 10, e exp eleva e à potência de seu argumento.

Como o PASCAL não possui operador de potenciação, evitam-se certas preposições absurdas muito comuns no BASIC, como:

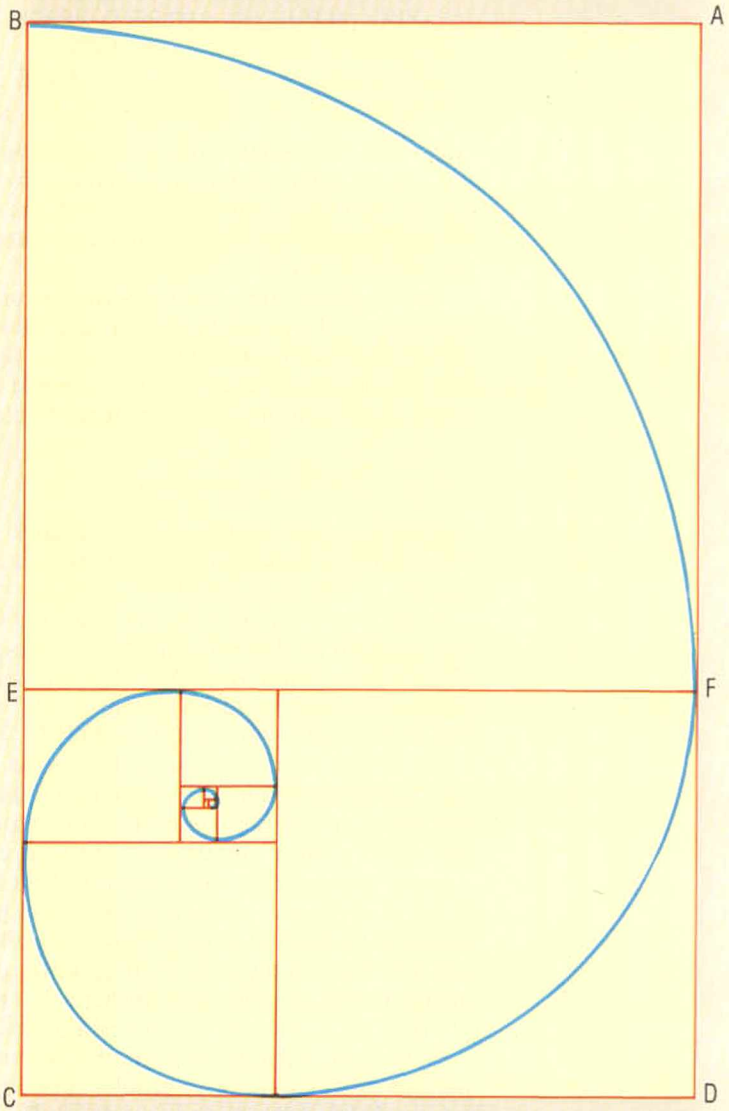
```
500 LET D=B^2+4*A*C
```

que calcula o quadrado de um número por um procedimento muito mais lento e impreciso do que a simples instrução B * B. Observe que a função SQR (N) do PASCAL fornece um valor in-

A divisão áurea

Um e somente um ponto corta uma linha em duas partes, de tal modo que a razão entre seus comprimentos é igual à razão entre a parte maior e a linha inteira. Essa razão — chamada áurea — é um número irracional como o pi, com dígitos até o infinito.

A divisão áurea, usada pelos artistas e arquitetos gregos, é freqüentemente retomada nas teorias da proporção, mas suas manifestações mais comuns se encontram na natureza, como em conchas marinhas: os vetores de seus raios, separados em 90°, correspondem à razão áurea.



As arestas do retângulo ABCD respeitam a razão áurea. Traçando-se um quadrado ABEF, o novo retângulo EFDC tem arestas na mesma razão que ABCD. Repetindo-se o processo, os vértices dos quadrados continuam sendo pontos da "espiral áurea".

teger para o quadrado de N (supondo-se que N seja integer), enquanto SQR (X/3) forneceria um valor real. Outra diferença do PASCAL em relação ao BASIC: emprega-se a função SQRT (ALGO), e não SQR (ALGO), para se obter a raiz quadrada de um número.

Tipos de subfaixa

Cada tipo escalar possui uma faixa (range) definida e ordenada de valores, podendo ser usado como um tipo "principal" do qual derivamos tipos secundários definidos por subfaixas (subranges) de valores. Os limites inferior e superior de cada subfaixa são indicados por dois pontos (..) na definição do identificador do tipo de subfaixa; por exemplo:

```
TYPE
  BYTE = 0..255; {SUBRANGE DE INTEGER}
  ALFA = 'A'..'Z'; {SUBRANGE DE CHAR}
  COR = (VERMELHO, AMARELO, VERDE,
    BRANCO, PRETO); {NOVO TIPO}
  B e P = BRANCO..PRETO; {SUBRANGE DE
    COR}
```

Além das vantagens citadas, um compilador PASCAL de bom rendimento armazenará variáveis do tipo BYTE, por exemplo, em 8 bits em lugar dos 32 possíveis (se for usada a representação integer de 4 bytes).

Assim, uma grande matriz desse tipo ocupará apenas 25% da memória exigida para valores integer não qualificados. Observe que o tipo B e P possui dois valores possíveis, de modo análogo à seguinte definição dos valores booleanos:

```
TYPE
  BOOLEAN = (FALSE, TRUE);
```

Todos os tipos de subfaixa assumem tanto a classificação dos dados como as operações definidas em seu tipo principal.

A definição anterior de ALFA admite outros valores além das letras do alfabeto. Em especial, o conjunto de caracteres ASCII possui seis símbolos adicionais no intervalo entre "Z" e "a", inclusive os símbolos de colchetes ("[" e "]"). Entre os símbolos reservados do PASCAL, esses são usados para delimitar os componentes de alguns tipos de dados estruturados, do mesmo modo que nas outras linguagens. O BASIC chegou a usar esses símbolos para indicar índices de matrizes, mas passou a empregar parênteses porque os computadores antigos às vezes possuíam códigos ASCII incompletos, (half-ASCII), que não incluíam letras minúsculas, colchetes etc. (O Apple II original também sofria dessa deficiência.)

O PASCAL permite as seguintes alternativas para os casos em que esse problema ocorre: (. e .) em vez de [e], e (* e *) no lugar de {e}. Para delimitadores de comentários, a segunda opção é bastante comum; no entanto, apenas os compiladores de padrão ISO permitem a substituição das chaves. Apenas uma ou as duas substituições podem ser feitas, de modo que: (* ESTE E UM COMENTARIO VALIDO}.

Função	Valor resultante
ABS (K)	Valor absoluto de K
SQR (K)	Quadrado de K
SQRT (K)	Raiz quadrada de K
SIN (A)	Seno de A radianos
COS (A)	Co-seno de A radianos
ARCTAN (T)	Ângulo com a tangente T
LN (K)	Logaritmo natural de K
EXP (L)	"e" elevado à potência L

Programa Áureo

O programa Áureo, abaixo, produz termos de Fibonacci e os imprime juntamente com sua razão. Ele revela uma das propriedades interessantes da série de Fibonacci (1, 2, 3, 5, 8, 13 etc.), a de que cada par sucessivo de termos se aproxima gradativamente do valor da divisão áurea.
O programa fornece exemplos adicionais da estrutura REPEAT do PASCAL. Como exercício, tente modificar a condição de encerramento do loop para que ele se interrompa quando o próximo termo de Fibonacci a ser calculado exceder MAXINT.

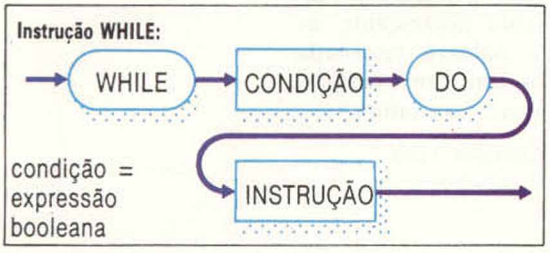
```
PROGRAM      AUREO      ( OUTPUT );
CONST
      EPSILON      = 1.0E-08;
TYPE
      FIBONACCI    = 1 .. MAXINT;
VAR
      PRIMEIRO,
      SEGUNDO,
      PROXIMO      : FIBONACCI;
      RAZAO,
      OURO         : REAL;
      CONTADOR     : INTEGER;
BEGIN
  WRITELN ( 'A DIVISAO AUREA' : 30 );
  WRITELN;
  WRITELN ( 'SERIE DE FIBONACCI:' );
  PRIMEIRO := 1; { POR DEFINICAO }
  SEGUNDO := 1;
  RAZAO := PRIMEIRO/SEGUNDO;
  CONTADOR := 0;
  REPEAT
    IF CONTADOR MOD 10 = 0 THEN
      BEGIN { CABECALHO A CADA 10 LINHAS }
        WRITELN;
        WRITELN ( 'PRIMEIRO' : 10, 'SEGUNDO' : 10,
          'RAZAO' : 14 );
        WRITELN;
      END;
    WRITELN ( PRIMEIRO : 10, SEGUNDO : 10,
              RAZAO : 16 : 8 );
    CONTADOR := CONTADOR + 1;
    OURO := RAZAO; { SALVA A RAZAO
      ANTERIOR }
    PROXIMO := PRIMEIRO + SEGUNDO;
    PRIMEIRO := SEGUNDO { AVANCO }
    SEGUNDO := PROXIMO; { A SERIE }
    RAZAO := PRIMEIRO / SEGUNDO
  UNTIL ABS ( RAZAO - OURO ) < EPSILON;
  WRITELN;
  WRITELN ( 'A RELACAO AUREA E : ',
        100 * RAZAO : 10 : 5, '%' );
END.
```

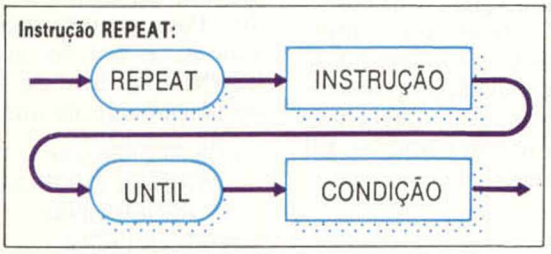
Estruturas iterativas

Além das estruturas seqüenciais, que empregam BEGIN e END, e da escolha ou seleção, que utiliza IF e CASE, o PASCAL dispõe de um terceiro tipo de instrução: a "iteração", ou repetição. A instrução FOR resulta num loop "controlado por contador", enquanto as instruções WHILE e REPEAT são loops "controlados por condições".

Em primeiro lugar, calcula-se a expressão booleana delimitada pelas palavras reservadas WHILE e DO. Se o resultado for verdadeiro, a instrução seguinte do PASCAL (qualquer uma, mesmo as estruturadas, independentemente de sua complexidade) será executada repetidas vezes, enquanto a condição permanecer verdadeira.

Essa condição booleana é reavaliada após cada execução da instrução no "núcleo" da estrutura WHILE. Isso significa que pelo menos um





dos itens avaliados na expressão deve receber alguma modificação pelas atividades no interior do loop. Por exemplo, a instrução

```
WHILE MAXINT > 1 DO
  WRITELN ('EM LOOP!')
```

teria grande dificuldade em se encerrar. Eis um exemplo prático (e mais confiável):

```
READ (RETIRADA);
WHILE RETIRADA > SALDO DO
  BEGIN
    WRITELN ('SEM FUNDOS, TENTE
      NOVAMENTE');
    WRITE ('QUANTIA?');
    READ (RETIRADA)
  END
```

A mais importante vantagem da estrutura WHILE é bem exemplificada por esse fragmento de programa. Se, quando for lido pela primeira vez, o montante sacado estiver dentro do limite do saldo atual da conta corrente, o loop WHILE não será executado. Se o núcleo for executado, o loop será repetido até que a expressão se torne falsa. O fluxo de controle retornará à operação seqüencial, e a primeira instrução após a estrutura WHILE será então executada. Assim, para realizarmos uma iteração em que apenas uma condição é verdadeira, usamos o loop WHILE.

Muitas vezes ocorrem situações em que precisamos executar o núcleo de um loop pelo menos uma vez para estabelecer a condição que o interromperá, como acontece no exemplo do programa Áureo. Nessas circunstâncias, a estrutura REPEAT-UNTIL traduz o algoritmo com maior precisão e fornece uma sintaxe natural de alto nível.

Observe que a condição de encerramento agora aparece no final da estrutura e que ela é oposta, em termos lógicos, à condição usada para codificar a estrutura WHILE. Isso significa que o loop REPEAT se encerrará quando a expressão booleana resultar verdadeira, enquanto uma estrutura WHILE se interromperá (ou sequer se iniciará) se a condição for falsa.

Se o núcleo da instrução REPEAT contiver instruções, elas deverão ser escritas entre as palavras delimitadoras BEGIN-END, como uma instrução composta; mas essa regra não é estrita, porque as palavras reservadas REPEAT e UNTIL delimitam adequadamente o núcleo do loop. Nada impede que se introduzam as palavras supérfluas BEGIN e END (procedimento parecido com o uso de ponto-e-vírgula antes de uma palavra reservada), embora isso não crie nenhuma instrução nula.

O fragmento abaixo conta a freqüência da letra "e" numa sentença fornecida por meio do teclado. Você deverá finalizar a sentença com um ponto quando o loop REPEAT se encerrar, e o resultado será apresentado.

```
CONTADOR:=0;
REPEAT
  READ (SIMBOLO);
  IF SIMBOLO='E' THEN
    CONTADOR:=CONTADOR + 1
UNTIL SIMBOLO='.';
WRITELN ('EXISTEM', CONTADOR: 1,
  ' "E"S NA SENTENCA.')
```

Nesse caso, a prioridade de qualquer das estruturas não é muito acentuada, e um loop WHILE poderia ser igualmente usado:

```
CONTADOR:=0;
READ (SIMBOLO);
WHILE SIMBOLO< > '.' DO
  BEGIN
    IF SIMBOLO='E' THEN
      CONTADOR:=SUCC (CONTADOR);
    READ (SIMBOLO)
  END;
WRITE ('EXISTEM', CONTADOR : 1,);
WRITELN (' "E"S NA SENTENCA.')
```

É fácil perceber a diferença entre as duas estruturas. O procedimento necessário para estabelecer a condição de encerramento (a instrução READ, no caso) em geral ocorrerá duas vezes com o uso da estrutura WHILE: primeiro, antes

da entrada para a estrutura; depois, como a última instrução no núcleo do loop WHILE.

Embora o loop WHILE seja a única estrutura essencial em qualquer linguagem, convém ter um meio de repetir algo um número especificado de vezes, talvez numa certa faixa de valores no interior de uma escala. O PASCAL possui uma terceira estrutura para criar esses loops "controlados por contador". Nesse ponto, o programador em BASIC se sentirá em terreno conhecido, mas há várias diferenças importantes entre a estrutura do PASCAL e o loop FOR do BASIC.

Em primeiro lugar, qualquer variável escalar pode ser usada como controlador do loop e não apenas as variáveis integer. Além disso, a instrução FOR-DO do PASCAL é completamente confiável; tal como o loop WHILE, ela não poderá ser executada de forma alguma (se, por exemplo, o valor inicial for maior que o final), e há restrições drásticas quanto ao uso da variável controladora no PASCAL. Sobretudo, não é permitida a modificação desse valor em qualquer ponto do núcleo do loop — uma precaução muito eficaz quando se lida com procedimentos e funções. Eis um exemplo de estrutura ilegítima:

```
FOR N:=1 TO 10 DO
  IF N=10 THEN
    N:=1
```

Isso não faz sentido, pois, apesar de se determinar que o loop seja executado dez vezes, a instrução IF restabelecerá o valor 1 para N, criando assim um loop infinito.

Observe que a sintaxe da estrutura FOR-DO possui uma instrução de atribuição entre os dois primeiros delimitadores (FOR e TO), que atribui o valor inicial à variável de controle do loop. Já o valor final da escala é dado pela expressão escrita entre as palavras reservadas TO e DO. Esses dois valores devem ser do mesmo tipo escalar simples que o controlador. Eis outros exemplos:

- FOR LETRA:='A' TO 'Z' DO {ETC}
- FOR MES:=JAN TO DEZ DO {ETC}
- FOR N:=N TO SUCC (MAXINT DIV 1000) DO {ETC}

Resposta do exercício áureo

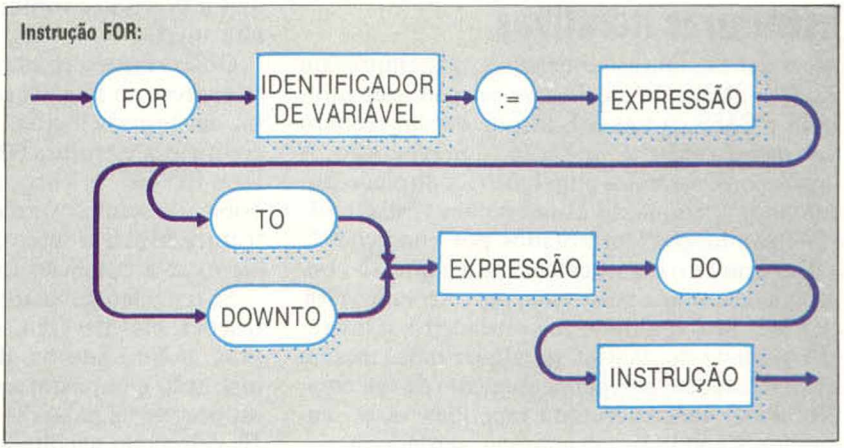
Com a representação do complemento dos pares, se um resultado integer ultrapassar MAXINT, o bit de sinal ficará "ligado". O número então aparece negativo; desse modo, nosso programa Áureo não funcionaria, já que o tipo Fibonacci exclui valores negativos. Mesmo com valores integer, o loop nunca se encerraria. Não podemos dizer:

```
UNTIL PRIMEIRO+SEGUNDO>MAXINT
```

pois, por definição, não há valor maior que MAXINT. Contudo, uma reestruturação engenhosa para efetuar a subtração em vez da adição permite resolver isso:

```
UNTIL SEGUNDO>MAXINT — PRIMEIRO
```

Agora a aritmética real e a constante épsilon já não são mais necessárias.



O último exemplo supõe que N tenha recebido um valor adequado em algum ponto anterior do programa, mas, se esse valor ultrapassar o valor final especificado, o loop não será executado. Para seguir uma escala decrescente de valores, é preciso usar a palavra reservada DOWNTO no lugar de TO. Assim, por exemplo, no lançamento de um foguete, usaríamos:

```
FOR REGRESSIVA:=10 DOWNTO 0 DO
  WRITELN (REGRESSIVA : 32-3*
    REGRESSIVA);
WRITELN ('FOGO!')
```

O PASCAL mantém um controle rigoroso do loop FOR e não admite qualquer equivalente do incremento ou decremento por meio do comando opcional STEP do BASIC.

```
PROGRAM      CERVEJA      (OUTPUT);

TYPE
  TIPO = (LEVE, AMARGA, ESPECIAL, PRETA);
VAR
    PRECO,
    GARRAFAS : REAL;
    CERV : TIPO;
BEGIN
  WRITELN ('GARRAFAS' : 11,
      'SUAVE' : 9, 'AMARGA' : 9,
        'ESPECIAL' : 9, 'PRETA' : 9);
  WRITELN;
  GARRAFAS := 0.5; {COMECA COM MEIA}

  REPEAT
    IF GARRAFAS - TRUNC (GARRAFAS) > 0.4
      THEN {IMPRIME COMO XX.X}
        WRITE (GARRAFA : 10 : 1)
      ELSE {IMPRIME COMO INTEIRO}
        WRITE (ROUND (GARRAFAS) : 7, ' ' : 3);

    FOR CERV := SUAVE TO PRETA DO
      BEGIN
        CASE CERV OF
          SUAVE : PRECO := 1000;
          AMARGA : PRECO := 1200;
          ESPECIAL : PRECO := 1500;
          PRETA : PRECO := 1300;
        END;
          {ARREDONDA E IMPRIME O PRECO}
        WRITE (ROUND (PRECO * GARRAFAS) : 9 : 2);
      END;
      {INICIA NOVA LINHA}
    WRITELN
    GARRAFAS := GARRAFAS + 0.5;

  UNTIL GARRAFAS > 10;
END.
```

Qual é a sua?

O programa Cerveja imprime uma lista com os preços de quatro tipos de cerveja, numa escala que aumenta em graus de meia garrafa. Cada tipo de cerveja é representado por um identificador de constantes conceituais enumeradas na definição TYPE. A seleção do preço é feita por uma instrução CASE. As garrafas inteiras são impressas em valores integer, com dois espaços adicionais, enquanto a formatação dos totais reais elimina casas decimais não desejadas.

WAFADRIVE

O aparecimento do sistema Microdrive da Sinclair estimulou companhias independentes a produzir equipamentos para armazenamento de dados, tais como o Wafadrive da Rotronics.

Uma vez conectado à CPU do Spectrum ou do TK 90X, o Wafadrive agrupa várias interfaces e o dispositivo de armazenamento, além de slots para a ligação de outros periféricos.

O nome dado a unidades desse tipo, "floppy tape", indica sua posição entre as fitas cassete e os discos flexíveis. O princípio de seu funcionamento é simples: o rolo de fita é contínuo, de modo que rapidamente se encontram os dados. E, como o diretório dos discos, o floppy tape mantém um índice de todos os programas e arquivos armazenados na fita. Dessa forma, uma lista do conteúdo está sempre à disposição do usuário.

O Wafadrive é envolto numa caixa de plástico preto com cabo flexível multivias finalizado por um conector que se encaixa no bus de expansão do microcomputador. Na parte frontal, há duas janelas para as unidades de minicartuchos. Entre as janelas estão três diodos de emissão de luz (LEDs). O LED central é o indicador de ligado/desligado; os outros dois indicam se a unidade está em funcionamento.

Fácil acesso
Ao contrário do Microdrive da Sinclair, o Wafadrive vem num estojo único com dois acionadores de fita, uma saída serial RS232 e uma interface paralela Centronics. O cabo flexível adapta-se ao conector edge e com isso a unidade pode ficar logo atrás do teclado, facilitando o acesso.

Isso significa que o Wafadrive agrupa numa só unidade várias interfaces e o dispositivo de armazenamento. Possui também slots para ligação de outros periféricos.

Atrás da unidade existem três conectores tipo edge. À esquerda há um para o bus paralelo. O conector edge central é uma interface compatível com o padrão Centronics e dá saída para uma impressora paralela. O terceiro corresponde a uma saída serial padrão RS232 que possibilita a utilização de modems e outros dispositivos seriais.

Os minicartuchos projetados especificamente para o Wafadrive assemelham-se aos usados nos Microdrives da Sinclair. Dentro do cartucho existe uma fita contínua do tipo usado para videocassete, com largura de 1,8 mm. Ela é usada no lugar das fitas de áudio convencionais porque tem maior resistência e maior capacidade de armazenamento de informações. Uma vez formatada, a fita pode conter até 128 Kbytes de dados.

Os cartuchos são quase duas vezes mais largos que os produzidos pela Sinclair, se bem que de comprimento e espessura similares. Não precisam ser acondicionados em caixas protetoras, pois uma cobertura deslizante automática, parecida com a existente nos minidisquetes de 3 1/2 polegadas da Sony, reveste a fita. No lado esquerdo do cartucho há uma plaquinha de proteção contra gravação.

Os comandos do Wafadrive são bem semelhantes aos usados pelo Microdrive da Sinclair. Nos dois sistemas o comando é seguido por um *, como em SAVE *, LOAD * e VERIFY *, indicando que o dispositivo de armazenamento externo deve ser acessado. Quando se formata um cartucho da Sinclair usa-se o comando FORMAT 'm';0;"nome", onde 'm';0 se refere ao número do Microdrive em operação. Quando se usa o Wafadrive, o comando é FORMAT 'a:nome', onde a: se refere ao nome da unidade em uso. Com o Wafadrive, só pode haver a designação a: ou b:; no Microdrive o número pode variar de zero a sete.

O sistema stream

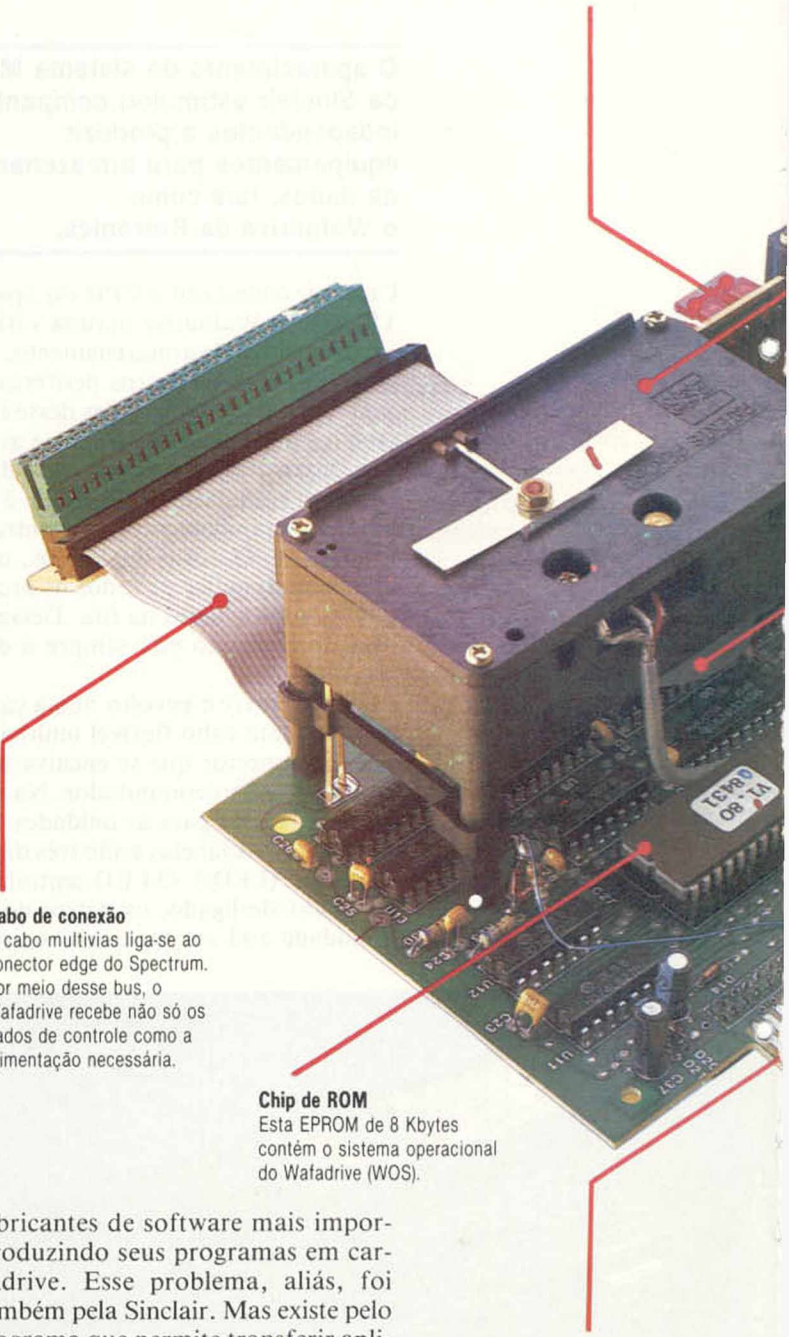
O Wafadrive também tira vantagem do sistema stream usado no Spectrum, no qual existem dezesseis canais reservados para controle de entrada/saída. Alguns desses streams são ocupados pela tela e pela impressora. Os canais de números 4 a 15 ficam disponíveis para outros periféricos e os streams de saída para o Wafadrive são acessados pelo comando OPEN #. O Wafadrive também adiciona dois streams extras ao sistema: os canais R e C, reservados para a saída serial RS232, e a paralela Centronics, respectivamente.

Uma ROM de 8 Kbytes, na placa principal, contém os comandos ampliados do BASIC usados para o controle do sistema. Esse sistema de operação do Wafadrive (WOS, Wafadrive Operating System) funciona paginando os 8 Kbytes inferiores da ROM do Spectrum. O comando LOAD *, por exemplo, gera um erro; dessa forma, quando o interpretador BASIC encontra tal comando na tela, chama uma rotina que se encarregue desse erro. O comando, no entanto, é interceptado pelo WOS, que aciona a ROM do Wafadrive. Esta, por sua vez, assume a manipulação do erro e interpreta LOAD * como comando.

Comparado com o Microdrive, o Wafadrive da Rotronics é um pouco lento. Um Microdrive de 100 Kbytes requer, em média, 3,5 segundos para localizar uma informação, transferida para o computador a uma velocidade de até 19,2 Kbauds. Por outro lado, o Wafadrive só atinge uma velocidade de transferência de 18 Kbauds, com tempo máximo de acesso de 45 segundos num cartucho de 128 Kbytes. Essa menor rapidez é parcialmente compensada por sua mais alta confiabilidade.

Embora esses tempos de acesso sejam muito menores do que os obtidos em fita cassete, são ainda muito maiores que o tempo de acesso das unidades de disco. No entanto, os floppy tapes incorporam um procedimento útil para acessar o catálogo da fita, equivalente ao diretório dos discos e disquetes. O catálogo é armazenado no primeiro setor do cartucho, após a emenda que une as duas extremidades da fita. Desse modo, para se obter o catálogo de um cartucho, o acionador tem de girar a fita até encontrar a emenda e ler o próximo setor. Depois de vários segundos, o cabeçote da fita passa o setor do catálogo. No entanto, se o comando CATALOGUE for usado de novo, o acionador, ao invés de girar a fita inteira mais uma vez, move-se apenas por uma fração de segundo e mostra o mesmo catálogo de novo — uma vez chamado, ele se mantém na RAM.

Portanto, o WOS apenas verifica se é o mesmo cartucho que está inserido, examinando o setor seguinte. Em caso afirmativo, o WOS mostrará o catálogo que ele já incorporou à RAM.



LEDs de controle das unidades
Indicam ao usuário que unidade está sendo acessada.

Cabo de conexão
O cabo multivias liga-se ao conector edge do Spectrum. Por meio desse bus, o Wafadrive recebe não só os dados de controle como a alimentação necessária.

Chip de ROM
Esta EPROM de 8 Kbytes contém o sistema operacional do Wafadrive (WOS).

Conector paralelo
Este conector edge permite que outras interfaces compatíveis com o Spectrum sejam adicionadas ao sistema.

Um pequeno preço?

Seguindo a linha de implementar um sistema de armazenamento de dados que permita o uso do Spectrum e seus compatíveis para aplicações profissionais, a Rotronics inclui na venda do Wafadrive o programa de processamento de texto Spectral Writer. Este é um sistema abrangente, que utiliza o Wafadrive por completo. Funções como a inserção de palavras e a eliminação de linhas inteiras são chamadas usando-se a tecla [Shift].

O que faz o sucesso ou o fracasso de qualquer meio de armazenamento num computador é o suporte de software. Isso ainda constitui séria desvantagem para o Wafadrive, uma vez que nenhum dos fabricantes de software mais importantes está produzindo seus programas em cartuchos Wafadrive. Esse problema, aliás, foi enfrentado também pela Sinclair. Mas existe pelo menos um programa que permite transferir aplicativos comerciais para cartuchos Wafadrive.

Os usuários são forçados a comprar o cassete comercial e o cartucho para transferi-lo, mas este talvez seja um preço pequeno para se obter tempos de acesso muito menores.

Outra dificuldade do Wafadrive está nos conectores edge, na parte traseira da máquina. Como eles não são do tipo padrão, os usuários precisam adaptar as próprias interfaces. Se não, devem procurar periféricos com conexões apropriadas.

SCOPUS

HÁ 10 ANOS DESENVOLVENDO TECNOLOGIA BRASILEIRA

No Brasil, a era da informática começou recentemente. Mas não para a SCOPUS que, desde 1975, trabalha em projetos, produções, criações e fabricações das mais avançadas, destacando-se no mercado porque aposta no talento e capacidade de técnicos, engenheiros e especialistas brasileiros. Não mede esforços. Sempre à procura de soluções modernas e criativas, desenvolve o que há de mais arrojado em matéria de informática. Sem copiar o que é feito fora. Conheça agora algumas das atuais conquistas tecnológicas da SCOPUS.

Tecnologia e sofistic

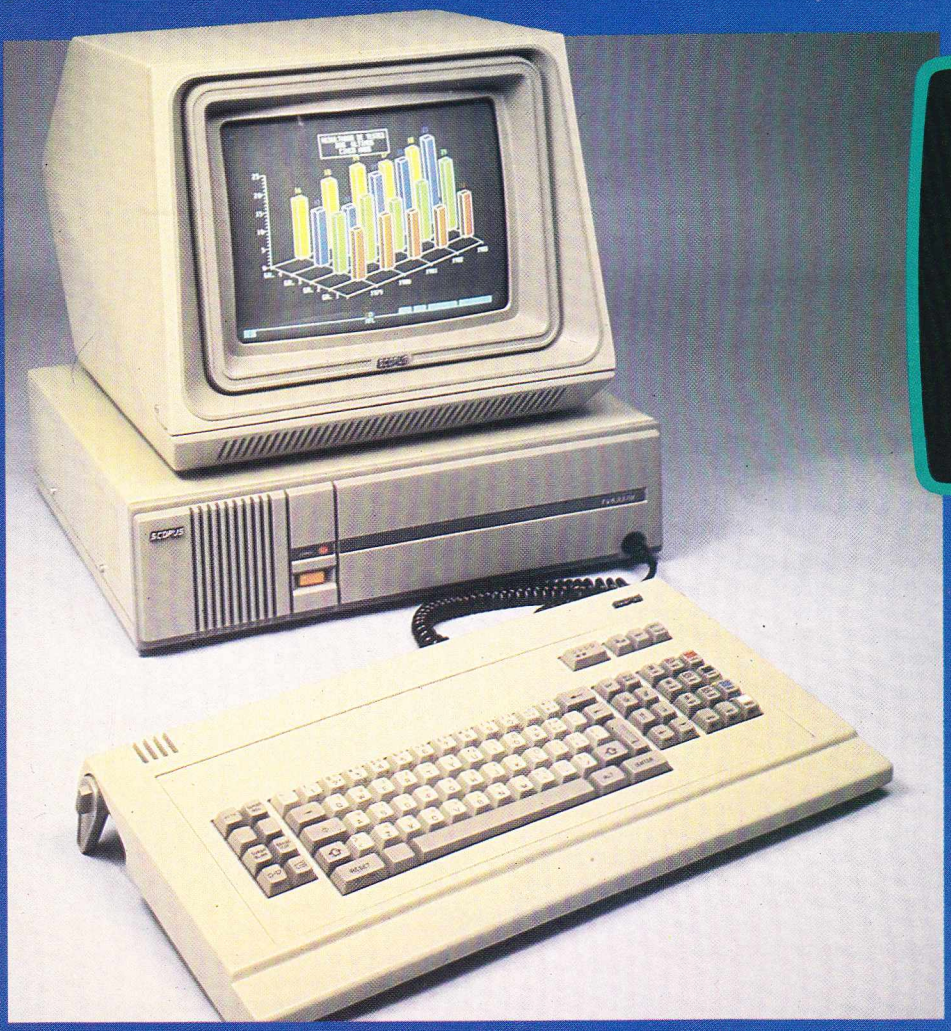
NEXUS 1684

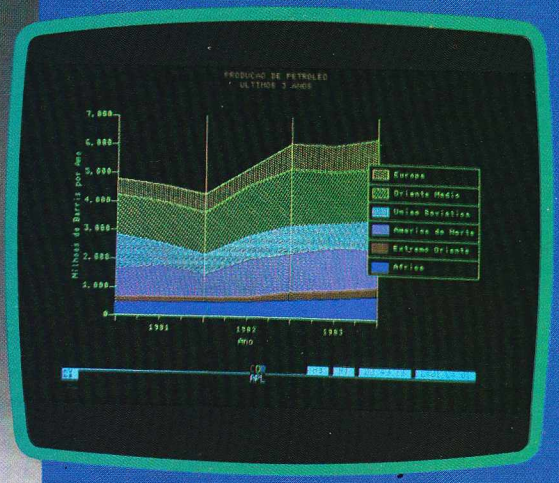
Como o mais recente integrante da família NEXUS 1600 de microcomputadores, o modelo 1684 traz uma importante novidade: a integração da unidade de disco magnético "Winchester", capaz de armazenar 10 milhões de caracteres (10 Mbytes). Dirigido a aplicações profissionais, o NEXUS pode ser utilizado para diversas operações, tais como análise financeira, análise de orçamento, balancete, entre outras tarefas gerenciais e administrativas. Ele se integra também a vários sistemas de grande porte, como computadores IBM, Burroughs ou Honeywell. Oferece características exclusivas, como o "clock" de CPU de 8 MHz e o tratamento de vídeo sem cintilação. Por trabalhar junto com a unidade Winchester, foi possível obter-se uma grande redução de custos além de um melhor aproveitamento do espaço físico.

TVA 3379

Com um hardware sofisticadíssimo e tecnologia (brasileira, é claro) das mais avançadas, o terminal de vídeo TVA 3379 tem como proposta principal facilitar a interpretação de dados fornecidos pelo computador através de gráficos a cores. É o único terminal desse tipo disponível para sistemas IBM existente no mercado. Usando o recurso das cores, o TVA 3379 facilita a leitura e inter-

ção em três versões.





pretação de qualquer informação, o que antes só poderia ser obtido através de tabelas numéricas. É a SCOPUS atenta a todo avanço e tendência do mercado mundial, facilitando a relação entre o homem e a máquina. Apenas usando o talento, o esforço e a criatividade de centenas de engenheiros brasileiros.

PC 2140

O mais novo e econômico "tradutor de linguagens", criado e desenvolvido com tecnologia totalmente nacional, o processador de comunicação PC 2140 permite a conexão simultânea de 32 equipamentos (terminais ou microcomputadores) com um computador de grande porte, através de uma única linha BSC-3. Isto permite o acesso ao banco de dados dos sistemas de grande porte, como IBM ou Burroughs, possibilitando ao usuário levantar importantes informações disponíveis no computador, entre elas, saldos de conta bancária, consultas ao pregão da Bolsa de Valores e verificação do estoque. O PC 2140 pode ser ligado a Unidades de Controle de Transmissão IBM 270X, 370X e ICA. A ligação dos equipamentos ao PC 2140 deve ser direta ou via modem, podendo-se também utilizar o modem de resposta automática.

Qualidade e assistência. A Scopus garante.

A avançada tecnologia do NEXUS 1600, PC 2140 e TVA 3379 já garante ao consumidor alto grau de qualidade e confiabilidade. Mas não é só isso que a SCOPUS oferece. Todos os serviços de manutenção são prestados por equipes de profissionais de alto nível técnico, em constante treinamento, que atendem rapidamente a qualquer chamado. Sempre investindo no aperfeiçoamento de sua estrutura de atendimento, a SCOPUS garante a mais elevada disponibilidade a seus equipamentos e a certeza de uma assistência técnica permanente.

DADOS TÉCNICOS

NEXUS 1600

UNIDADE DE SISTEMA

Placa de sistema: Microprocessador Intel 8088 (4,77 ou 8 MHz) • 64 Kbytes de memória EPROM • 256 Kbytes de memória RAM • 2 interfaces de comunicação serial: assíncrona e síncrona/assíncrona, padrão RS-232C, em velocidades até 19200 bauds • 5 conectores para placas de expansão.

Placa controladora de disco: controla duas unidades de disco flexível de 5 1/4", possui relógio-calendário de tempo real não volátil e duas interfaces para "joy-sticks".

Placa controladora de vídeo: modo alfanumérico: telas de 40x25 ou 80x25 caracteres, 16 cores • modo gráfico: telas de 320x200 pontos a 4 cores ou 640x200 pontos monocromáticas, interface para "light-pen".

Placa controladora de Winchester: controla a unidade de disco rígido tipo Winchester.

Unidade de disco flexível de 5 1/4": um acionador com cabeça dupla, densidade dupla, com capacidade de 320 Kbytes.

Unidade de disco rígido tipo Winchester: um acionador com capacidade de 10 Mbytes, slim.

MONITOR DE VÍDEO

Monocromático, 12", fósforo verde, interface padrão RGB, média resolução e configuração multinível, reproduzindo a informação da cor em tons de intensidades variáveis. Opcionalmente, pode-se configurar o Nexus 1600 com um monitor de vídeo cromático, 14", média resolução.

TECLADO

Tipo máquina de escrever, 86 teclas.

SOFTWARE BÁSICO

O Sistema Operacional SISNE, desenvolvido pela Scopus, emula o sistema MS-DOS 1.1 do microcomputador PC-IBM e contém todos os utilitários compatíveis com os utilitários do MS-DOS 1.1. Possui uma interface homem-máquina bastante poderosa e gerencia, com máxima eficiência, os recursos físicos do equipamento.

MÓDULOS DE EXPANSÃO

N1664: expansão de memória RAM (448 Kbytes) • **N1667**: co-processador CP-8087 • **N1669**: interface paralela para impressora.

MÓDULOS PARA COMUNICAÇÃO DE DADOS

Sistema Incox: interface coaxial para conexão a redes IBM 3270 • **NX-3270**: módulo para comunicação com sistemas IBM sob protocolo BSC-3 • **NX-830**: módulo para comunicação com sistemas Burroughs sob protocolo Poll-Select • **NX-VIP**: módulo para comunicação com sistemas Honeywell Bull sob protocolo VIP 7700 • **NX-Z**: módulo para comunicação assíncrona serial de dados.

TVA 3379

MÓDULOS

Unidade de vídeo: apresenta tela útil de 1920 ou 2560 posições, respectivamente para os modelos 2 ou 3. Possui duas linhas adicionais de informações no vídeo para mensagens do sistema e de erros, além de possibilitar a apresentação de cronômetro e indicador de linha e coluna. O monitor é cromático, de alta resolução. O cinescópio, de 14 polegadas, pode ser configurado para que se apague automaticamente, evitando a queima desnecessária do fósforo.

Unidade lógica: comporta as placas de controle do terminal, de geração de caracteres plano triplo, e as interfaces de comunicação com o vídeo, unidade controladora e teclado.

Teclado: comunica-se com a unidade lógica de forma serial e permite a conexão de um dispositivo auxiliar de entrada. Está disponível em quatro opções: APL Português, APL Inglês, "Typewriter" Português e "Typewriter" Inglês, todas com bloco destacado de teclas de cor e de símbolos programáveis.

PC 2140

CARACTERÍSTICAS FUNCIONAIS

Interface de comunicação com terminais ou micros: serial, assíncrona, código ASCII, full duplex, padrão elétrico RS-232C.

Placa interface de comunicação: cada placa permite que até 8 dispositivos sejam conectados ao PC 2140 com seleção de velocidades independentes.

Conexão de modem: a interface de comunicação com os terminais ou micros permite a ligação de modem, podendo ser constante ou por meio de linhas discadas, através de centrais telefônicas.

Interface de comunicação com o computador: serial, síncrona, sob o protocolo BSC-3, padrão elétrico RS-232C, em velocidades até 9600 bps. Emula as unidades de controle tipo IBM 3271 ou 3274 com 32 terminais IBM 3277 ou 3278, modelo 1.

Conversão de códigos: adotados pelos terminais (ASCII) em códigos adotados pelo computador (EBCDIC).

Execução de comandos: permitem ao computador ativar funções como, por exemplo, estabelecer conexões com os terminais, desfazer conexões, transmitir textos e informar o estado de ligações.

Verificação de paridade: em todas as mensagens com o computador, o PC 2140 gera e verifica os caracteres de controle de bloco (CRC-16), pedindo retransmissão automática em caso de erro.

MODELOS

O PC 2140 está disponível em quatro opções, que diferem apenas em relação ao número de placas de controle dos terminais.

PC 2140/8: controla até 8 terminais (1 placa).

PC 2140/16: controla até 16 terminais (2 placas)

PC 2140/24: controla até 24 terminais (3 placas)

PC 2140/32: controla até 32 terminais (4 placas)

CENTROS DE ASSISTÊNCIA TÉCNICA

BELÉM: Tel.: (091) 228-0011 - ramal 242
BELO HORIZONTE: Tel.: (031) 224-8457
BRASÍLIA: Tels.: (061) 224-9856 e 224-9835
CAMPINAS: Tel.: (0192) 42-7333 - BIP: 657
CUIABÁ: Tels.: (065) 322-0542 321-5792 - BIP: 139
CURITIBA: Tel.: (041) 262-5211 - BIP: 150
FORTALEZA: Tel.: (085) 224-7999 - BIP: DA2
LONDRINA: Tel.: (0432) 22-0558
PORTO ALEGRE: Tel.: (0512) 32-1200 - BIP: 798
RECIFE: Tel.: (081) 221-3100 - BIP: 765
RIBEIRÃO PRETO: Tels.: (016) 636-3967 634-3331/3509 - BIP: 76B
RIO DE JANEIRO: Tels.: (021) 240-4925 e 240-6114
SALVADOR: Tel.: (071) 233-4117 - BIP: D12
SÃO JOSÉ DOS CAMPOS: Tels.: (0123) 22-8247 21-2004 - BIP: 536
SÃO PAULO: Tel.: (011) 255-7744

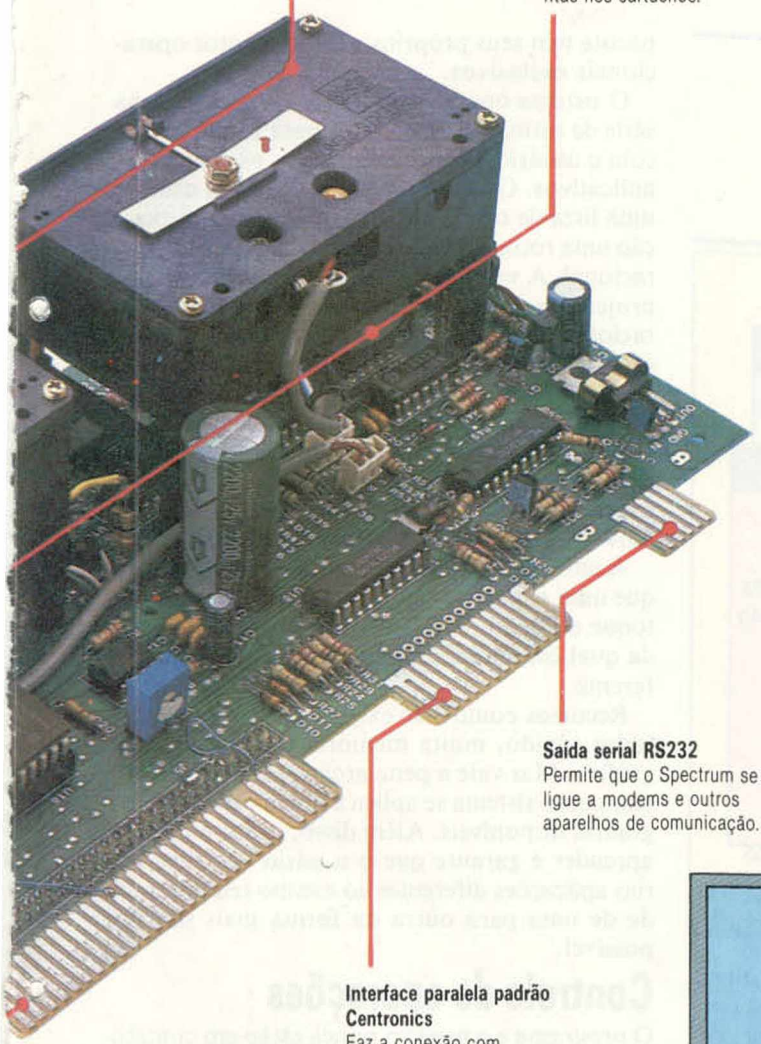
SCOPUS tecnologia s.a.

Filiada à ABICOMP

As especificações contidas neste folheto podem ser alteradas pelo fabricante sem aviso prévio.

Acionadores de fita
O Wafadrive possui duas unidades auto-suficientes.

Motores
Cada unidade tem seu motor elétrico próprio para girar as fitas nos cartuchos.



Saída serial RS232
Permite que o Spectrum se ligue a modems e outros aparelhos de comunicação.

Interface paralela padrão Centronics
Faz a conexão com impressoras.

WAFADRIVE

DIMENSÕES
230 x 110 x 180 mm.

INTERFACES
Saída serial RS232, interface paralela padrão Centronics, conector edge do Spectrum.

FORMATO
Minicartuchos flexíveis com fita em anel sem fim.

CAPACIDADE
Acham-se disponíveis cartuchos de 16, 64 e 128 Kbytes.

VELOCIDADE DE TRANSFERÊNCIA
16 Kbauds.

TEMPO MÁXIMO DE ACESSO
6,5 s (16 K), 45 s (128 K).

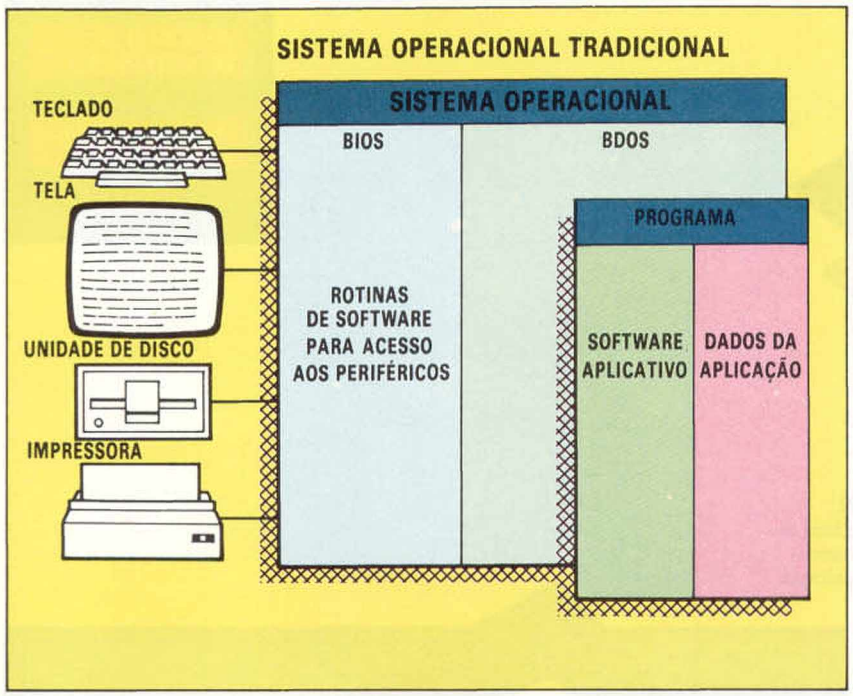
O minicartucho do Wafadrive tem quase o dobro do tamanho do equivalente do Microdrive da Sinclair. O cartucho assemelha-se a um cassete convencional em suas proporções, mas a fita interna forma um anel sem fim. Isso significa que a fita não precisa ser girada para trás a fim de acessar os dados que já tenham passado pelo cabeçote de leitura e gravação.

PAGINAÇÃO

O termo inglês "paging" traduz-se por "paginação" e indica a divisão de programas ou dados em blocos de tamanho fixo, carregados na memória quando necessário.

CONTROLE TOTAL

A integração mais comum produz programas que abrangem todas as funções de que você necessita. Contudo, esse não é o melhor sistema, pois tais programas são enormes e usam muita memória.



Operando sob ordens
Com o sistema operacional tradicional, o programa em execução tem o controle completo. Sua lógica determina o que aparece na tela, quando é preciso acessar os discos, e o modo de ler o teclado. Suas instruções gerais são transferidas para o sistema operacional, que lida com a manipulação detalhada do hardware em uso. A execução do programa predomina e a subordinação do sistema operacional está pressuposta.

A abordagem alternativa para a questão da integração baseia-se num princípio que conta com o sistema operacional do computador para prover os recursos básicos da integração. Todos os programas escritos para esse sistema ajustam-se automaticamente a ele para trabalharem juntos.

A criação de um sistema operacional desse tipo não foi tarefa fácil, pois ele precisa de um computador com hardware e software muito mais sofisticados do que nos sistemas tradicionais. A Apple abriu o caminho com seus microprocessadores destinados ao Lisa e ao Macintosh e várias outras empresas estão desenvolvendo sistemas para ser rodados em microcomputadores como o IBM PC.

Os programas elaborados para os novos sistemas operacionais são bem diferentes dos desenvolvidos para os sistemas tradicionais. Boa parte desses programas inclui a interface com o usuário, ou seja, as rotinas que recebem comandos e informações do usuário e lhe apresentam os resultados. As opiniões quanto à maneira de operar programas divergem, de forma que cada pacote tem seus próprios procedimentos operacionais exclusivos.

O sistema operacional integrado oferece uma série de rotinas incorporadas para comunicação com o usuário, utilizável em todos os programas aplicativos. Quando precisa mostrar ao usuário uma lista de opções, o programa tem à disposição uma rotina pronta para isso, no sistema operacional. A vantagem é que todos os programas projetados para trabalhar com esse sistema operacional terão aproximadamente o mesmo procedimento. Depois que você tiver aprendido um dos programas do sistema, estará pronto para usar todos os outros à disposição.

Um modo de interagir com o usuário, específico desses programas, é o mouse, dispositivo usado para escolher opções na tela por meio de movimentos correspondentes do cursor.

A alternativa para o mouse é a tela digital, em que uma matriz de raios luminosos responde ao toque do dedo. A tela divide-se em janelas, cada qual contendo uma opção ou uma tarefa diferente.

Recursos como esse exigem um microprocessador rápido, muita memória e alta resolução gráfica. Mas vale a pena arcar com essas despesas, pois o sistema se aplica a quase todos os programas disponíveis. Além disso, é muito fácil de aprender e garante que o usuário visualize várias aplicações diferentes ao mesmo tempo e mude de uma para outra da forma mais simples possível.

Controle de operações

O programa e o usuário nunca estão em contato direto; é o sistema operacional que controla tudo, o tempo todo. Cada aplicativo torna-se uma extensão do sistema operacional e o computador fica sendo um único ambiente integrado.

Isso leva à segunda grande diferença na forma pela qual os sistemas funcionam. Na maneira tradicional, a comunicação entre o sistema operacional e o programa ocorre numa única direção. O programa determina que certa tarefa seja realizada e o sistema operacional a executa. Num processo integrado, o sistema operacional está no comando e atribui tarefas ao programa. O sistema operacional manda, por exemplo, esta mensagem ao programa: "O usuário acaba de mover seus dados para o outro lado da tela. Redesenhe a tela", ou "Pare tudo. O usuário moveu o mouse para uma aplicação diferente". É preciso que o programa seja capaz de responder às requisições do sistema operacional e vice-versa.

Uma vez atingido esse grau de cooperação entre todos os programas, é fácil construir um ambiente integrado. Cada programa tem sua própria janela na tela. Quando o usuário posiciona o mouse numa janela e faz uma opção, o sistema operacional notifica o programa específico e a operação se realiza.

Por exemplo: se o usuário muda o cursor para um canto de uma janela e seleciona a opção que transfere essa janela para nova posição, as rotinas do sistema operacional realizam a tarefa. E, se necessário, informam o programa sobre as mudanças, para que ele possa retificar apropriadamente sua tela. Se o usuário leva o mouse para uma janela diferente, o programa original torna-se inoperante e o sistema operacional começa a trabalhar com o novo programa. Portanto, mudar de aplicação é tão simples quanto mover o mouse.

Todos os programas que estiverem na tela ao mesmo tempo encontram-se na memória e disponíveis para uso. Para facilitar o processo, muitos sistemas têm memórias maciças — por exemplo: 1 Megabyte no Lisa e 512 Kbytes no Macintosh. Mesmo assim, o sistema operacional precisa ocasionalmente armazenar e acessar em disco informações e partes de programas, a fim de acomodar tudo. Para que o sistema se torne razoavelmente rápido, é necessário utilizar um disco rígido.

Os dados são facilmente permutados entre os programas, porque o sistema operacional dispõe de uma série incorporada de formatos e rotinas para transferência de dados. Quando se "exporta" uma informação de certo programa e se pede a outro que a "importe", o sistema operacional suspende o primeiro programa e aciona o segundo. Então, pede ao aplicativo corrente que leia e processe a informação que está vindo de outro programa.

Esses caminhos se estabelecem automaticamente, para que, quando você alterar dados numa planilha, por exemplo, um gráfico da mesma planilha também seja alterado.

Os dois programas não rodam concomitantemente; o sistema operacional apenas faz as trocas entre eles, sempre que necessário.

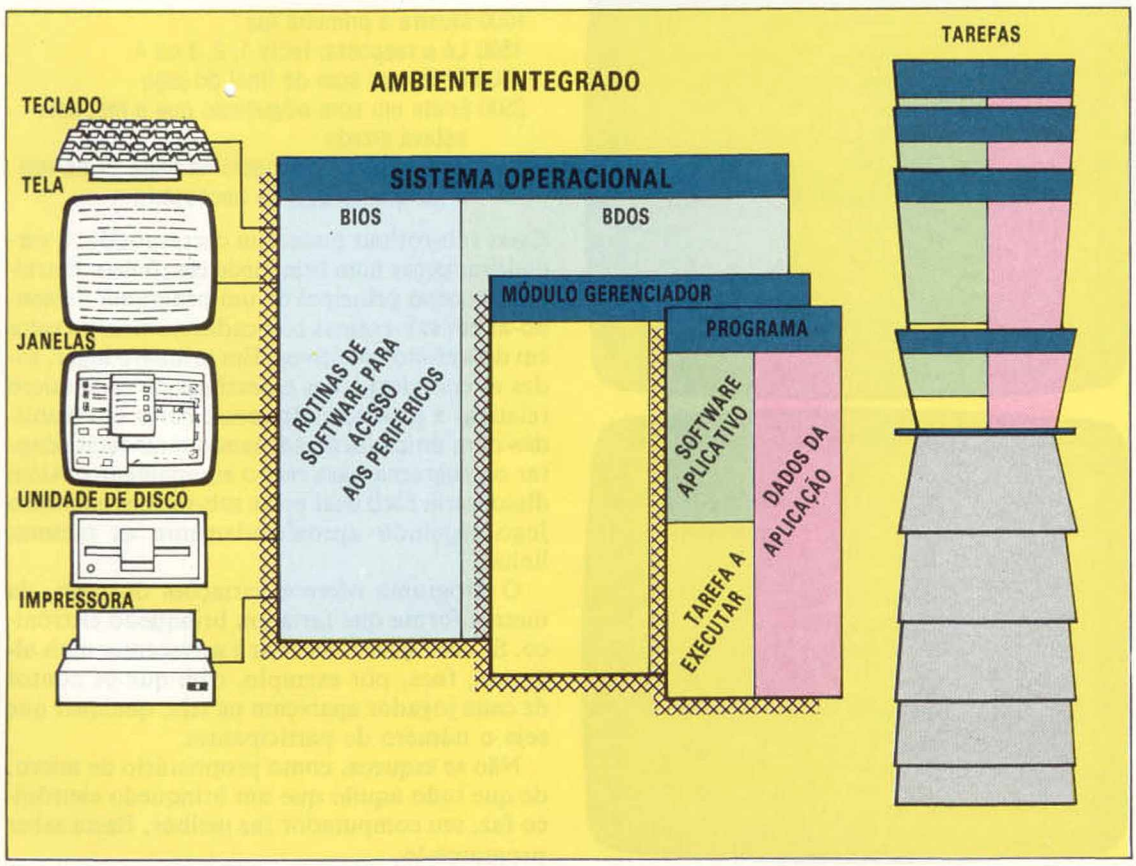
O Lisa da Apple apresenta um conceito mais sofisticado. Nessa máquina, a informação pode ser "recortada" de uma janela com qualquer programa e "colada" em outra. Os dados são transferidos junto com suas informações de formatação, de modo que um gráfico produzido num gerador de gráficos seja transferido para qualquer outro programa sob a forma de gráfico mesmo.

Essa é a forma mais sensata de integrar softwares. Ela permite unir e combinar quaisquer programas no sistema, mover-se de um programa para outro e transferir dados entre eles. A desvantagem é que esse sistema requer um hardware sofisticado — e muito caro. Além disso, há poucos programas disponíveis para a integração. Qualquer grande inovação tecnológica demora para se tornar comum. A tecnologia do mouse e das janelas, por exemplo, só apareceu no mercado dez anos depois de desenvolvida pelas equipes de pesquisa da Xerox.

SISTEMAS

BIOS (Basic Input/Output System): Elo de ligação lógica entre o hardware e o software aplicativo, que controla todas as operações de entrada e saída. Residente em ROM.

BDOS (Basic Disk Operating System): Núcleo do sistema operacional, que detém as funções de controle do disco.



Operações combinadas
Num sistema integrado, o sistema operacional é reforçado pela adição de um módulo de gerenciamento, que trata todos os programas e dados correntes como tarefas a serem programadas e executadas. O módulo interage com o sistema operacional como simples software de suporte de sistema. Tira e põe tarefas na memória principal e nos discos, de acordo com os pedidos do usuário e com as necessidades da tarefa atual. Está equipado para passar informações de uma para outra aplicação em formato padronizado e permite assim a transferência de dados entre as tarefas. O gerenciador é, ele mesmo, uma tarefa de alta prioridade.

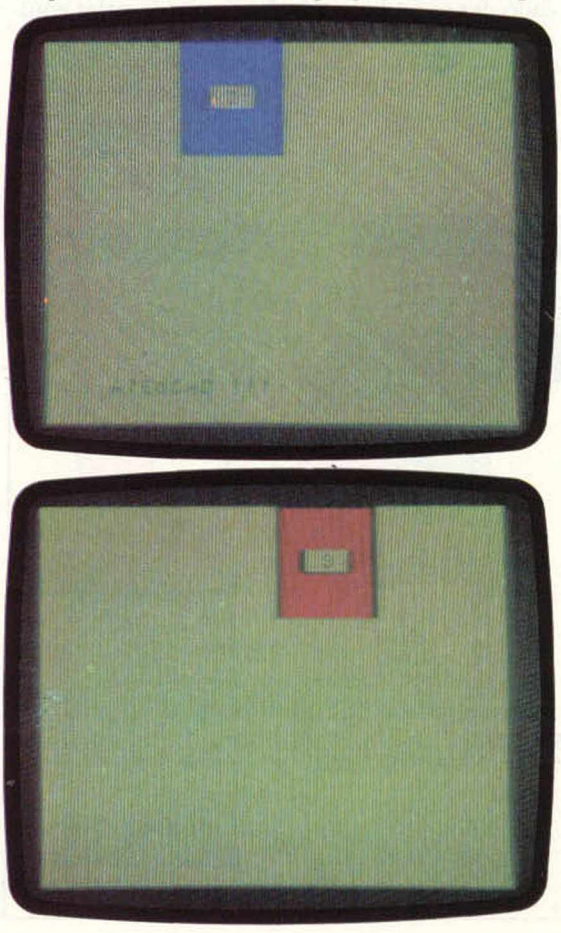
CORES E NÚMEROS

Tal como alguns brinquedos eletrônicos, o jogo aqui examinado baseia-se na antiga brincadeira Seu Mestre Mandou. O programa usa cores e sons para entreter por meio de um estimulante teste de memória.

No jogo conhecido como Seu Mestre Mandou, o líder dá instruções do tipo "Seu mestre mandou pôr as mãos na cabeça" ou "Seu mestre mandou ficar em pé" etc. Os jogadores serão desclassificados se obedecerem a uma instrução que não comece por "Seu mestre mandou".

Essa brincadeira de salão transformou-se num jogo eletrônico (no Brasil, o Genius, da Estrela), quando se desenvolveu uma série de brinquedos que utilizam microprocessadores para controlar um conjunto de botões, luzes e campainhas. Na versão computadorizada, a máquina é o líder.

O brinquedo eletrônico fornece uma seqüência de sons e cores e os jogadores devem repetila, pressionando as teclas apropriadas. A máquina então acrescenta outro som à seqüência. Esses jogos tornaram-se tão populares que muitos imaginam o Seu Mestre Mandou como sendo exclusivamente eletrônico.



Memória prodigiosa
As combinações de cores e sons geradas pela versão para micros do jogo Seu Mestre Mandou são praticamente inesgotáveis e ilustram a grande vantagem dos computadores sobre nós: eles nunca esquecem nada, ao contrário da limitada memória humana.

Se você possui um micro, a idéia de um brinquedo eletrônico que só joga Seu Mestre Mandou deve parecer muito estranha. Um brinquedo pode ser portátil, durável e fácil de usar, mas mesmo os melhores jogos logo se tornam tediosos. Já a capacidade do computador de executar centenas de programas diferentes assegura que ele nunca ficará desinteressante.

O programa que aqui listamos é o Repita!, que executa um jogo baseado em quatro cores, cada qual correspondendo a um som e a uma tecla (1, 2, 3 ou 4). Uma luz acende e ele pede para ser imitado. Se você acertar, o programa exibirá duas luzes, e assim por diante.

Há dois modos pelos quais você perde esse jogo: se demorar demais para pressionar a tecla correta ou se pressionar a tecla errada três vezes numa partida. Para tornar o desafio mais difícil, essa versão também vai ficando cada vez mais rápida, embora não haja necessidade de se repetir a seqüência na velocidade em que ela é executada pelo computador. O jogo tem um máximo de até cinqüenta luzes numa seqüência, mas é pouco provável que você alcance esse limite. A maior parte dos jogadores mal consegue chegar a quinze luzes.

Vale a pena estudar o modo como o programa é estruturado. Todos os comandos de cor e som estão agrupados em sub-rotinas no final do programa deste modo:

```
1000 Mostra a primeira luz
1500 Lê a resposta: tecla 1, 2, 3 ou 4
2000 Emite um som de final do jogo
2500 Emite um som advertindo que a resposta
     estava errada
6000 Mostra uma mensagem na linha 21 da tela
     e faz uma pausa, se necessário
```

Essas sub-rotinas poderiam corresponder a verdadeiras peças num brinquedo eletrônico. Extraí-las do corpo principal de um programa (isolando-as em sub-rotinas colocadas no final) resulta em dois efeitos positivos. Em primeiro lugar, todas as complexidades específicas de cada micro relativas à produção de sons e cores são mantidas num único local, tornando mais fácil adaptar o programa para outro equipamento. Além disso, seria fácil usar essas sub-rotinas em outro jogo seguindo aproximadamente as mesmas linhas.

O programa oferece variações do jogo, da mesma forma que faria um brinquedo eletrônico. Se você quiser inventar e acrescentar mais algumas, faça, por exemplo, com que os pontos de cada jogador apareçam na tela, qualquer que seja o número de participantes.

Não se esqueça, como proprietário de micro, de que tudo aquilo que um brinquedo eletrônico faz, seu computador faz melhor. Basta saber programá-lo.

Repita!

```
10   REM ********************************
20   REM **                              **
30   REM **           REPITA             **
40   REM **                              **
50   REM **       PARA O APPLE           **
60   REM **                              **
70   REM ********************************
80   REM
90   RAM **    DADOS PARA GERAR SONS     **
100  DATA 172,1,3,174,1,3,169,4,32,168,2
     52,173,48,192,232,208,253,136,208,2
     39,206,0,3,208,231,96
110  FOR Z=770 TO 795:READ BY:POKE Z,BY:
     NEXT:POKE 799,0:POKE 800,0
120  REM **      INICIALIZA O JOGO       **
130  DIM C(4),P(4),A(50):H=0:N=0:W=3
140  P(1)=5:P(2)=8:P(3)=12:P(4)=15
150  C(1)=1:C(2)=2:C(3)=4:C(4)=6
160  B$=CHR$(127):REM ** BLOCO CHEIO   **
170  HOME:PRINT TAB(10);"REPITA !":PRINT
     :PRINT
180  IF N>0 THEN PRINT:PRINT"VOCE CONSEG
     UIU "N" SEQUENCIAS"
190  IF N>H THEN PRINT:PRINT'ESTE E O NO
     VO RECORDE !":H=N
200  IF H>0 THEN PRINT:PRINT"O RECORDE A
     TUAL E "H" SEQUENCIAS"
210  PRINT:PRINT"TENTE REPETIR AS SERIES
     DE LUZ E SOM DO COMPUTADOR USANDO A
     S TECLAS DE 1 A 4"
220  PRINT:PRINT"USE J PARA JOGAR OU P P
     ARA PARAR"
230  GET A$:IF A$="" THEN 230
240  IF A$="P" THEN HOME:END
250  IF A$<>"J" THEN 230
260  HOME:PRINT TAB(10);"REPITA !"
270  FOR A=1 TO 4:GOSUB 1000:NEXT A
280  N=0:M=0
290  N=N+1
300  A(N)=INT(RND(1)*4)+1
310  IF M=W THEN GOSUB 2000:W$=STR$(W):M
     $="*"+W$+" RESPOSTAS ERRADAS":GOSUB
     6000:GOTO170
320  M$="*ATENCAO !":GOSUB 6000
330  FOR I=1 TO N
340  A=A(I):GOSUB 1000
350  FOR J=1 TO 100/N:NEXT J:NEXT I
360  M$="REPITA A SEQUENCIA":GOSUB 6000
370  I=1
380  GOSUB 1500:IF T=0 THEN GOSUB 2000:M
     $="*MUITO DEVAGAR":GOSUB 6000:GOTO
     170
390  IF A<>A(I) THEN M=M+1:GOSUB 2500:GO
     TO 310
400  I=I+1:IF I<=N THEN 380
410  IF N>50 THEN M$="GANHOU COM 50 SEQU
     ENCIAS !!!":GOSUB 6000:GOTO 170
420  M$="*PREPARE-SE PARA A PROXIMA":GOS
     UB 6000
430  GOTO 290
1000 REM ********************************
1010 REM **  ROTINA GERADORA DE LUZ      **
1020 REM ********************************
1030 COLOR=C(A)
1040 P=(A-1)*8+2
1050 FOR L=10 TO 14
1060 VTAB L:HTAB P:PRINT;
1070 IF L=12 THEN PRINT B$;B$;A;B$;B$
1080 IF L<>12 THEN PRINT B$;B$;B$;B$;B$;
1090 NEXT L
1100 REM **  ROTINA GERADORA DE SOM      **
1110 POKE 768,2:POKE 769,A*50:CALL 770
1120 VTAB 10:HTAB P:PRINT"      "
1130 VTAB11:HTABP:PRINT" ";B$;B$;B$;" ";
1140 VTAB12:HTABP:PRINT" ";B$;A;B$;" ";
1150 VTAB13:HTABP:PRINT" ";B$;B$;B$;" ";
1160 VTAB 14:HTAB P:PRINT"      "
1170 RETURN
1500 REM ********************************
1510 REM **    VERIFICA    TECLADO       **
1520 REM ********************************
1530 T=250
1540 GET A$:IFA$="" THEN 1540
1550 A=VAL(A$):IFA<0 OR A>4 THEN 1540
1560 IF A=0 THEN RETURN
1570 GOSUB 1000
1580 RETURN
2000 REM
2010 REM **   RUIDO  DO FIM DO JOGO     **
2020 REM
2030 FOR Z=1 TO 200 STEP 20:POKE 768,1:P
     OKE 769,Z:CALL 770:NEXT Z:RETURN
2500 REM
2510 REM **        AVISO DE ERRO         **
2520 REM
2530 POKE 768,1:POKE 769,0:CALL 770
6000 REM ********************************
6010 REM **        EXIBE MENSAGEM        **
6020 REM ********************************
6030 VTAB 21:PRINT M$
6040 FOR Z=1 TO 400:NEXT Z
6050 S$=" ":FOR Z=1 TO 6:S$=S$+S$:NEXT Z
6060 VTAB 21:PRINT S$
6070 RETURN
```

MUDANÇAS DE CÓDIGOS

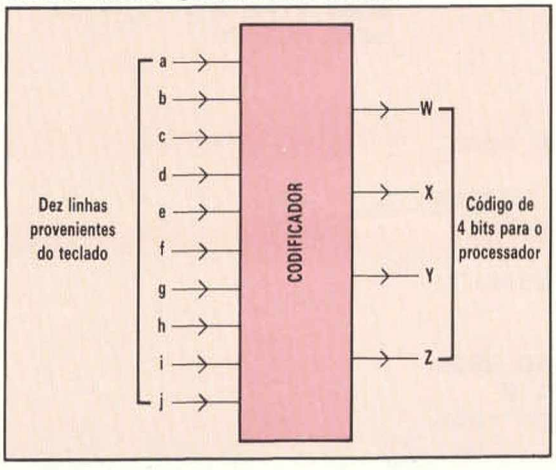
A CPU dos micros desempenha sua função pelo envio de instruções, sob forma de impulsos elétricos, para dispositivos internos ou periféricos, que as traduzem em sinais elétricos e vice-versa.

O processador envia instruções para os dispositivos internos (como o acumulador) ou para os periféricos do equipamento (como a impressora). Muitas vezes, reduz-se o número de linhas usado por um periférico a fim de fornecer uma entrada simplificada para o processador — é a codificação.

Um exemplo de codificador seria o circuito usado em conjunção com o teclado. Ele pode ter 64 linhas de saída, e uma delas produz sinal quando se pressiona a tecla correspondente. Como uma tecla é acionada a qualquer momento, cada um dos 64 sinais possíveis de saída pode ser codificado como um número binário de 6 bits. Isso significa que seis linhas bastam para transportar ao processador a indicação de que tecla foi pressionada. O dispositivo que converte 64 linhas em seis é o codificador.

Na prática, mais duas linhas são acrescentadas — uma para verificar a paridade e outra para indicar quando uma tecla [Shift] ou [Control] foi usada juntamente com outra.

Para demonstrar esse princípio, consideremos um teclado muito simples, que possua apenas dez teclas. Isso possibilita digitar um número de 0 a 9. Um código binário de 3 bits permite apenas oito combinações; assim, temos de desenhar um codificador para admitir dez linhas como entrada e fornecer quatro como saída.



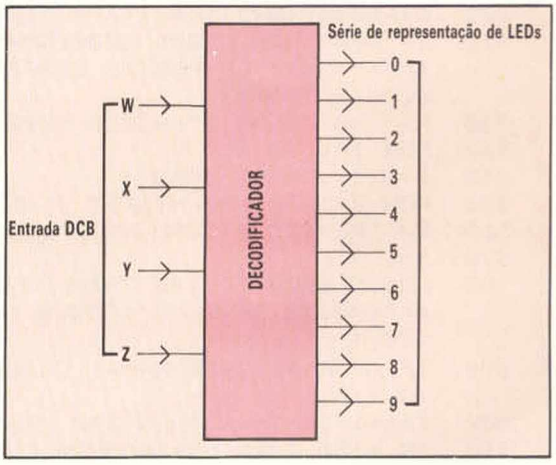
Como só uma das dez linhas pode ser ativada em dado momento, a tabela de validação para o codificador será:

Decimal	Entradas										Saídas			
	a	b	c	d	e	f	g	h	i	j	W	X	Y	Z
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	1	0	0	0	0	0	0	0	0	0	1	0
3	0	0	0	1	0	0	0	0	0	0	0	0	1	1
4	0	0	0	0	1	0	0	0	0	0	0	1	0	0
5	0	0	0	0	0	1	0	0	0	0	0	1	0	1
6	0	0	0	0	0	0	1	0	0	0	0	1	1	0
7	0	0	0	0	0	0	0	1	0	0	0	1	1	1
8	0	0	0	0	0	0	0	0	1	0	1	0	0	0
9	0	0	0	0	0	0	0	0	0	1	1	0	0	1

A decodificação é o inverso da codificação. Em lugar de admitir grande número de linhas de entrada para produzir poucas de saída, o decodificador admite pequeno número de entradas (em geral sob a forma de códigos binários provenientes do processador). A partir delas, seleciona uma entre as várias linhas de saída que controlam a atividade de um periférico (impressora ou traçador de gráficos tipo x-y, por exemplo).

Codificadores e decodificadores também são usados para controlar os movimentos da cabeça de disco e para selecionar canais de saída por meio dos números dos periféricos.

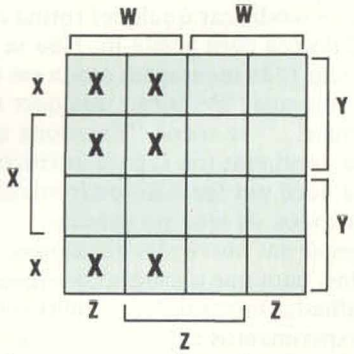
Vejamos como um decodificador simples pode ser desenhado com o uso das portas E, OU e NÃO, examinando o problema seguinte. É necessário um decodificador para converter decimais codificados em binários (DCB) numa forma que ligue uma das dez luzes (LED) correspondentes ao valor decimal do código. Aqui estamos lidando com um circuito que produz o inverso do exemplo do codificador.



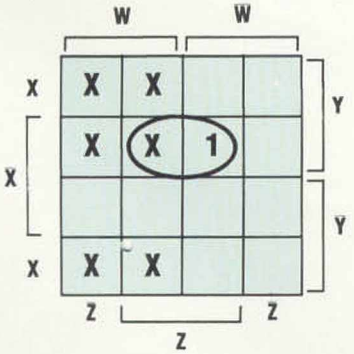
Os DCBs são representações de 4 bits dos dígitos decimais de 0 a 9; assim, o decodificador tem quatro linhas de entrada. Como qualquer combinação das quatro linhas pode ser ativada, há dezesseis entradas possíveis. Preservamos as dez primeiras combinações desse tipo e aplicamos

condições (X) de entrada não válida às outras. A tabela 1 é sua tabela de validação. A expressão booleana para cada saída parece exigir quatro termos (W, X, Y e Z), como na tabela 2.

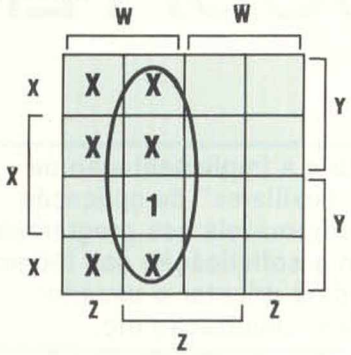
Como as condições de entrada não válida são as mesmas para cada uma das dez saídas, podemos unificar os termos para simplificá-los. Abaixo, o mapa de Karnaugh, com quatro variáveis, contém as seis condições de entrada não válida assinaladas.



Se examinarmos uma das saídas por vez, verificaremos que ainda podem ser feitas simplificações. Tomemos, por exemplo, a saída para o dígito 3. Pela colocação da expressão booleana no mapa-k, verificamos que é possível desenhar um loop que a inclua.



Assim, a expressão para a saída pode ser simplificada para $\overline{X}.Y.Z$. Tomando a saída 9 como um segundo exemplo, temos:



Pode-se desenhar um loop que utilize três das condições de entrada não válida e que represente a expressão booleana W.Z. Muitos dos outros termos de entrada fazem a simplificação de forma semelhante. Talvez você queira verificar se os termos da saída simplificada são os apresentados na tabela 3.

A tarefa agora é construir o diagrama de circuitos com as dez expressões booleanas. Como se usa cada entrada tanto na forma normal como na negativa, fica mais fácil construir o circuito de oito linhas paralelas representando esses termos. Cada entrada é formada desviando-se dessas linhas e usando-se portas E. Na parte inferior da página apresentamos o diagrama de circuito decodificador completo.

Exercício 6

1) Projete um codificador de três entradas que crie uma saída de 1 para entradas de 011, 101, 110 ou 111. A saída deverá ser 0 para outras combinações de entrada.

a) Desenhe a tabela de validação para o codificador.

b) Crie uma expressão booleana para a saída e simplifique-a.

c) Desenhe o circuito codificador.

Tabela 1

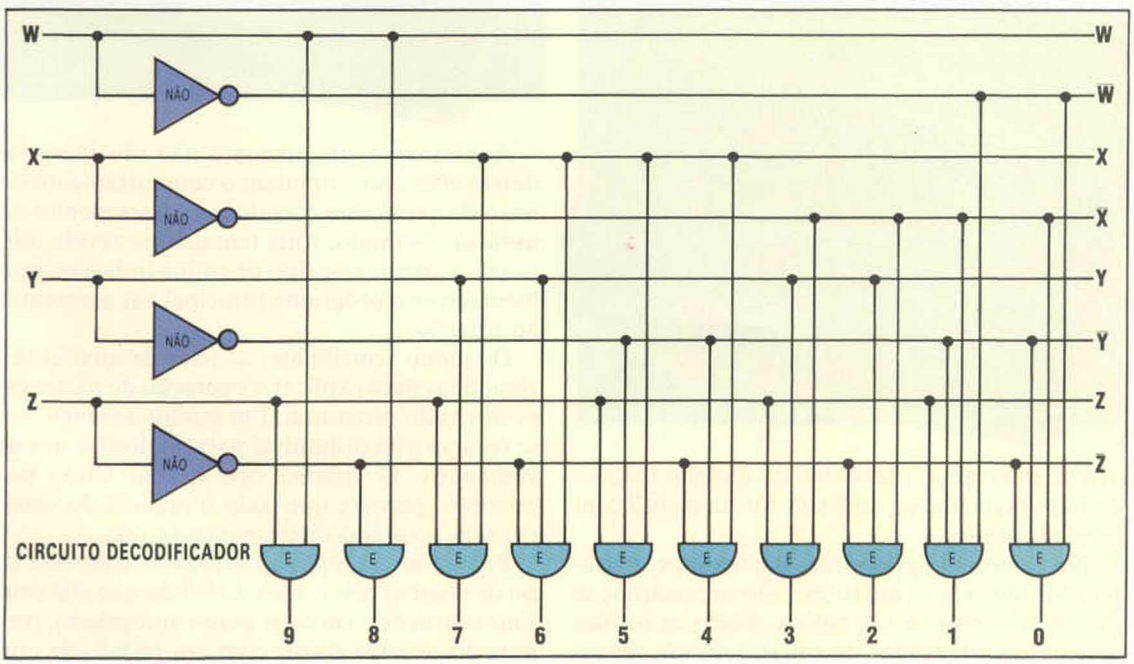
Entradas				Dig. DCB
W	X	Y	Z	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

Tabela 2

Digito DCB	Expressão booleana
0	$\overline{W}.\overline{X}.\overline{Y}.\overline{Z}$
1	$\overline{W}.\overline{X}.\overline{Y}.Z$
2	$\overline{W}.\overline{X}.Y.\overline{Z}$
3	$\overline{W}.\overline{X}.Y.Z$
4	$\overline{W}.X.\overline{Y}.\overline{Z}$
5	$\overline{W}.X.\overline{Y}.Z$
6	$\overline{W}.X.Y.\overline{Z}$
7	$\overline{W}.X.Y.Z$
8	$W.\overline{X}.\overline{Y}.\overline{Z}$
9	$W.\overline{X}.\overline{Y}.Z$

Tabela 3

Digito DCB	Expressão booleana
0	$\overline{W}.\overline{X}.\overline{Y}.\overline{Z}$
1	$\overline{W}.\overline{X}.\overline{Y}.Z$
2	$\overline{X}.Y.\overline{Z}$
3	$\overline{X}.Y.Z$
4	$X.\overline{Y}.\overline{Z}$
5	$X.\overline{Y}.Z$
6	$X.Y.\overline{Z}$
7	$X.Y.Z$
8	$W.\overline{Z}$
9	$W.Z$



AJUDA EM LINHA

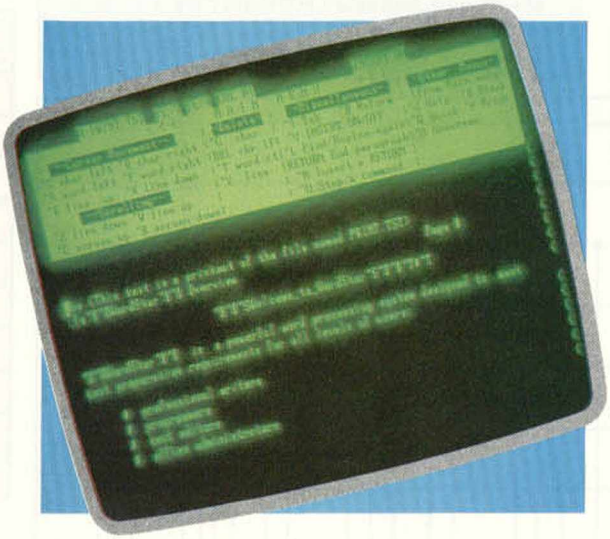
O projeto e a implementação de rotinas "auxiliares" de aplicação geral incorporáveis aos programas mostram a sofisticação das técnicas usadas para orientar o usuário: instruções, sinalização etc.

Os chips de memória se tornam progressivamente mais baratos e contêm cada vez mais informações. A próxima geração de computadores domésticos disporá de muito mais memória do que requerem nossos mais ambiciosos programas. Assim haverá espaço suficiente para melhores instruções, mensagens de erro e auxílio on-line ao usuário.

Três tipos principais de apoio ao usuário podem ser supridos dentro de um programa: instruções, telas de auxílio e sinalização.

As instruções tomam duas formas: são fornecidas num único bloco no começo do programa ou apresentadas conforme requisitadas no decorrer dele (por exemplo, mensagens para a entrada de dados pelo usuário). Idealmente, ambas devem estar disponíveis.

Em sua forma mais simples, as instruções compõem uma ou várias telas de texto, em linguagem corrente, sobre como usar o programa. O texto pode ser armazenado em strings (variáveis alfanuméricas) ou nos comandos DATA, dentro do programa, e será exibido quando requisitado por chamada a uma sub-rotina escrita com esse propósito.

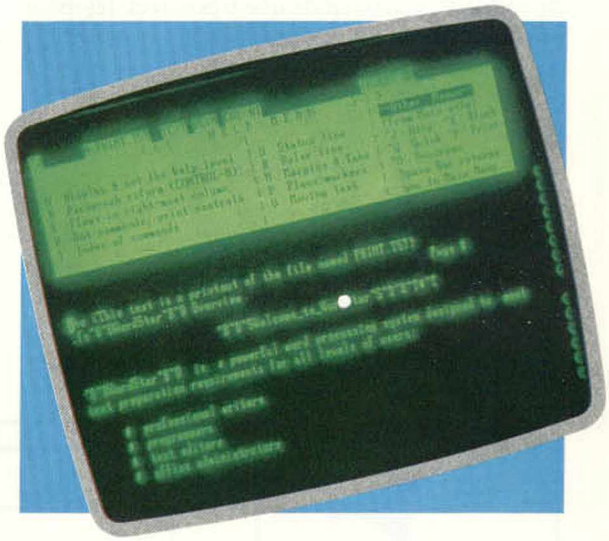


Auxílio permanente
O processador de texto WordStar da Micropro fornece um exemplo de grande aceitação de software guiado por comandos com auxílio on-line. O menu de ajuda que fica na tela pode ser abreviado ou eliminado pelo usuário, mas uma estrutura de telas de auxílio bem detalhadas está sempre disponível.

No começo do programa principal, pergunta-se ao usuário se as instruções são necessárias; se forem, chama-se a sub-rotina. Todas as rotinas que aceitam entradas do usuário devem ser escritas de forma que uma entrada específica (? ou |) chame a sub-rotina das instruções.

Convém padronizar o comando que mostra as instruções e modificar qualquer rotina de entrada da biblioteca para aceitá-lo. Não se esqueça de modificar toda mensagem usada em sua rotina, de forma que "Pressione qualquer tecla para continuar..." se torne "Pressione qualquer tecla para continuar (ou |, para instruções)". Isso dará a você um formato padronizado, utilizável em todos os seus programas.

Incluem-se nas instruções diagramas, detalhes e exemplos, para que o usuário pratique. Instruções detalhadas mostram-se comuns em programas de experimentos científicos — nos quais se pede ao usuário para executar uma tarefa específica, de determinado nível de habilidade, antes que lhe seja permitido entrar no programa principal.



As rotinas de treinamento não são fáceis de desenvolver, pois simulam o comportamento do resto do programa e avaliam o desempenho do usuário. Contudo, toda tentativa se revela útil, já que projetar esse tipo de rotina indica os problemas que o programa principal vai apresentar ao usuário.

De modo semelhante, as telas de auxílio são chamadas para explicar a operação de partes específicas do programa. Em muitos sistemas, esse recurso está disponível para explicar o uso de comandos. O sistema operacional Unix, por exemplo, permite que todo o manual do usuário seja acessado diretamente na tela.

Prover seus próprios programas com esse tipo de recurso não é mais difícil do que elaborar suas instruções: em cada ponto apropriado, permita ao usuário entrar com um pedido de aju-

da. Quando isso acontecer, o programa deve chamar a rotina de auxílio adequada.

Um programa complexo exige muitas telas de auxílio, sendo então necessário que se crie uma rotina geral com tal fim. No caso, o usuário fornece um número para identificar a tela solicitada. Em sistemas que disponham de disquetes, as telas de auxílio são armazenadas como arquivos separados. A rotina de auxílio acessa, então, o arquivo apropriado, a partir da entrada do usuário, e mostra seu texto na tela.

Quando longos, os textos de auxílio ou de instruções podem ocupar mais do que uma tela. Se for o caso, a rotina precisa ser projetada de maneira que o usuário possa acessar, de acordo com sua necessidade, telas mais à frente e mais atrás.

Você também deve assegurar-se de que o usuário possa abandonar a rotina em qualquer estágio e retornar ao ponto exato em que o programa principal foi deixado.

Cabe à rotina de auxílio fazer uma assinalação (flag) que informe à rotina de chamada que ela precisa voltar à última instrução antes do pedido de ajuda, e zerar, então, o flag.

Uma metáfora comum para a interação do usuário com um programa complexo é imaginar que ele navega por uma emaranhada rede de lógica. Principiantes no programa não entendem sua estrutura e podem ficar desnorteados. Assim sendo, a sinalização é necessária para orientar o usuário. Um menu é o exemplo mais claro: funciona como uma placa de trânsito que mostra as saídas possíveis a partir de um cruzamento.

Algumas instruções são mais importantes do que outras. Num sistema baseado em comandos, existem dezenas de comandos possíveis. No entanto, nem todos são relevantes, ou mesmo possíveis, estando-se em determinado ponto do programa. Se o número de opções for pequeno, convém mostrar uma ou duas linhas para explicar o que faz cada opção.

Algumas opções — como a de encerrar o programa — devem estar sempre disponíveis ou mesmo em exibição permanente. É conveniente também que os comandos para gravação, correção e outros, de aplicação específica, estejam constantemente disponíveis. Uma técnica comum consiste em programar tais comandos nas teclas de função e exibir uma mensagem de uma só linha mostrando a função de cada tecla. Indicar a possibilidade de uma saída para fora do programa dá segurança a quem o esteja utilizando pela primeira vez.

Boa administração

O Manager, sistema gerenciador do micro Apricot guia o usuário na escolha dos programas utilitários apresentados em menus. O auxílio é uma opção permanente, e consiste numa explicação dos outros itens. Esse é um bom exemplo de software orientado por menus, com extensos arquivos de auxílio.

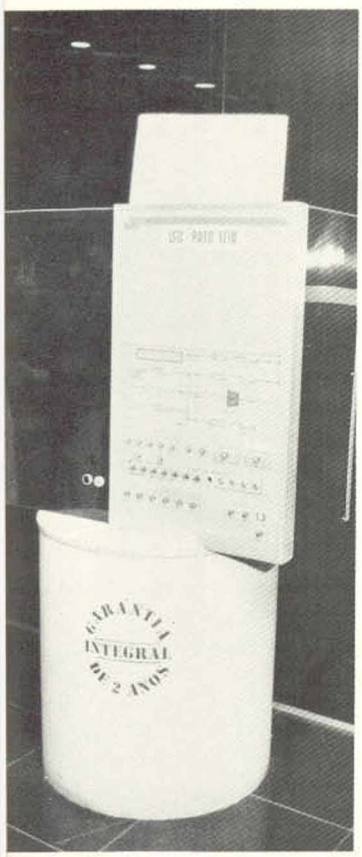
Alguns sistemas experimentais são capazes de monitorar o desempenho do usuário e ajustar o nível de auxílio fornecido de acordo com o desempenho. Se o programa pede o nome do usuário a cada vez que roda, pode-se manter um arquivo sempre atualizado dos usuários e seus níveis de habilidade.

Esses níveis são calculados a partir do número de vezes em que cada um deles executou o programa ou pelo maior número de pontos conseguidos, se o programa em questão for um jogo. À medida que aumenta o nível de habilidade, o tipo de ajuda e de sinalização muda, tornando-se mais breve e interferindo menos na execução do programa. Pode-se, também, pedir ao usuário que escolha o nível de auxílio, como no processador de texto WordStar.

A inclusão de instruções de auxílio pode ser um método valioso para melhorar o desempenho do programa. Uma vez projetada a rotina de auxílio, fica fácil modificá-la para que grave quais instruções foram requisitadas e com que freqüência. Isso dá clara indicação dos pontos problemáticos no programa.

OBJETIVO BRASILEIRO

Diversificação
Da linha de produção da Scopus, instalada na fábrica da foto, saem computadores tanto de uso pessoal como para aplicações profissionais, além de terminais de vídeo e de software especializado.



Pioneirismo
O primeiro computador projetado no Brasil foi o Patinho Feio, montado em 1972 pela equipe do Laboratório de Sistemas Digitais da USP, da qual participaram os fundadores da Scopus.

O desenvolvimento de tecnologia própria tem sido a principal preocupação da Scopus desde sua origem. Em 1983, ela entrou no promissor mercado dos microcomputadores de 16 bits.

Em 1975, Josef Manasterski, Célio Yoshiyuki e Edson Fregni — professores da Escola Politécnica da Universidade de São Paulo (USP) — fundaram a Scopus, que iniciou suas atividades como prestadora de serviços, tanto na manutenção de computadores como na pesquisa e projetos de informática. Mas logo ela passou a produzir, em pequena escala, equipamentos digitais para aplicações específicas.

Experiência não faltava aos três professores: além de participarem dos projetos, desenvolvidos em 1971-72 na USP para nacionalizar a tecnologia dos computadores, um deles — Manasterski — foi, no período 1973-75, o coordenador da equipe que criou o primeiro computador de médio porte da Politécnica, o G-10.

No início de 1985, todos os equipamentos produzidos pela Scopus eram inteiramente projetados por seus engenheiros e analistas, dos quais noventa dedicados à pesquisa e ao desenvolvimento de hardware e software.

Essa trajetória, marcada pela independência tecnológica, teve início em 1976, com o lançamento de um terminal não inteligente para computadores de médio e grande porte. Ele serviu de base para a criação do terminal inteligente TVA 800, compatível com a família de computadores fabricados pela Burroughs.

Ate 1981, esse equipamento foi vendido apenas a fabricantes (nacionais e estrangeiros) de sistemas de computação. Com a decisão de investir em terminais de vídeo compatíveis com os equipamentos IBM e em microcomputadores, a Scopus passou a comercializar seus produtos diretamente com o usuário final.

A estratégia da Scopus para ampliar sua participação no mercado envolveu duas áreas prioritárias: o desenvolvimento de software para redes de processamento de dados — como o sistema de gerenciamento de dados SBD/TS — e a exploração do segmento dos microcomputadores profissionais.

A empresa paulista, que já vinha produzindo a linha Microscopus de 8 bits desde 1981, lançou em 1983 o Nexus 1600, acompanhando a tendência mundial que privilegia os micros de 16 bits e sua conexão aos grandes sistemas. O Nexus 1600 integra-se à maioria dos sistemas de grande porte e às redes de serviços públicos, como Renpac, Aruanda e Cirandão.

```
      TO 119,149:HPLOT 159,139 TO 159,149
      :RETURN
  140 HPLOT 19,39 TO 49,49 TO 49,139:HPLO
      T 19,149 TO 19,39:HPLOT 19,139 TO 4
      9,139:HPLOT 19,49 TO 49,49:RETURN
  145 HPLOT 119,59 TO 159,59 TO 159,129 T
      O 119,129 TO 119,59 TO 129,69 TO 14
      9,69 TO 149,119 TO 129,119 TO 129,6
      9:HPLOT 149,69 TO 159,59:HPLOT 149,
      119 TO 159,129:HPLOT 129,119 TO 119
      ,129:RETURN
  150 HPLOT 229,49 TO 259,39 TO 259,149:
      HPLOT 229,139 TO 229,49:HPLOT 229,4
      9 TO 259,49:HPLOT 229,139 TO 259,13
      9:RETURN
  151 REM *                                 *
  152 REM * GERA PASSAGENS ENTRE SALAS     *
  153 REM *                                 *
  155 DIM S$(6,6)
  165 FOR A=1 TO 5:FOR X=1 TO 5:FOR Y=1
      TO 5
  167 & T10 * A + 10 * X + 10 * Y,10
  170 IF A <> 5 AND RND(1) < .80 THEN S$(
      X,A)=S$(X,A)+"0":GOTO 180
  175 S$(X,A)=S$(X,A)+"X"
  180 IF MID$(S$(X,A-1),(Y-1)*6+1,1)="0"
      THEN S$(X,A)=S$(X,A)+"0":GOTO 190
  185 S$(X,A)=S$(X,A)+"X"
  190 IF Y-2 < 0 THEN 200
  195 IF MID$(S$(X,A),(Y-2)*6+4,1)="0"
      THEN S$(X,A)=S$(X,A)+"0":GOTO 205
  200 S$(X,A)=S$(X,A)+"X"
  205 IF Y <> 5 AND RND(1) < .8  THEN S$(
      X,A)=S$(X,A)+"0":GOTO 215
  210 S$(X,A)=S$(X,A)+"X"
  215 IF X <> 5 AND RND(1) < .8  THEN S$(
      X,A)=S$(X,A)+"0":GOTO 225
  220 S$(X,A)=S$(X,A)+"X"
  225 IF MID$(S$(X-1,A),(Y-1)*6+5,1)="0"
      THEN S$(X,A)=S$(X,A)+"0":GOTO 235
  230 S$(X,A)=S$(X,A)+"X"
  235 NEXT:NEXT:NEXT
  240 X = INT (RND (1) * 3) + 2  :
      Y = INT (RND (1) * 3) + 2  :
      A = INT (RND (1) * 3) + 2
  245 RD = INT (RND (1) * 6) + 1: ON RD
      GOTO 250,255,260,265,270,275
  250 A=5:P1$=LEFT$(S$(X,A),(Y-1)*6):L=29
      -LEN(P1$):P2$=RIGHT$(S$(X,A),L):S$(
      X,A)=P1$+"0"+P2$:GOTO280
  255 A=1:P1$=LEFT$(S$(X,A),(Y-1)*6+1):L=
      29-LEN(P1$):P2$=RIGHT$(S$(X,A),L):
      S$(X,A)=P1$+"0"+P2$:GOTO 280
  260 Y=5:P1$=LEFT$(S$(X,A),(Y-1)*6+3):L=
      29-LEN(P1$):P2$=RIGHT$(S$(X,A),L):
      S$(X,A)=P1$+"0"+P2$:GOTO 280
  265 Y=1:P1$=LEFT$(S$(X,A),(Y-1)*6+2):L=
      29-LEN(P1$):P2$=RIGHT$(S$(X,A),L):
      S$(X,A)=P1$+"0"+P2$:GOTO 280
  270 X=5:P1$=LEFT$(S$(X,A),(Y-1)*6+4):L=
      29-LEN(P1$):P2$=RIGHT$(S$(X,A),L):
      S$(X,A)=P1$+"0"+P2$:GOTO 280
  275 X=1:P1$=LEFT$(S$(X,A),(Y-1)*6+5):L=
      29-LEN(P1$):P2$=RIGHT$(S$(X,A),L):
      S$(X,A)=P1$+"0"+P2$:GOTO 280
  280 SX=X:SY=Y:SA=A
  290 VTAB 23:PRINT "QUALQUER TECLA PARA
      COMECAR"
  300 IF PEEK(-16384) < 127 THEN 300
  310 POKE -16368,0
 1000 X = INT (RND (1) * 5) + 1  :
      Y = INT (RND (1) * 5) + 1  :
      A = INT (RND (1) * 5) + 1  :
      FC=1:GOTO 1220
 1010 HOME:VTAB 22:HTAB 18:PRINT FC$(FC)
      :A$="":D=0:IF LS=1 THEN PRINT X,Y,A
 1020 VTAB 22:PRINT "TEMPO :";T:FOR TEMPO
      =1 TO 80
 1025 IF PEEK(-16384) > 127 THEN 1030
 1027 NEXT:T=T+1:VTAB 22:PRINT "TEMPO :";
      T:GOTO 1020
 1030 GET A$
 1035 IF A$ = "*" THEN LS = 1
 1040 IF A$ = "F" THEN 1300
 1050 IF A$ = "C" THEN D = 1
 1060 IF A$ = "B" THEN D = 2
 1070 IF A$ = "N" THEN D = 3
 1080 IF A$ = "S" THEN D = 4
 1090 IF A$ = "L" THEN D = 5
 1100 IF A$ = "?" THEN 1290
 1110 IF A$ = "O" THEN D = 6
 1120 IF A$ = "V" THEN 1280
 1130 IF D = 0 THEN 1010
 1135 T=T+1
 1140 IF MID$(S$(X,A),(Y-1)*6+D,1) <> "0"
      THEN PRINT CHR$(7): GOTO 1010
 1150 ON D GOTO 1160,1170,1180,1190,1200,
      1210
 1160 A=A+1:GOTO 1220
 1170 A=A-1:GOTO 1220
 1180 Y=Y-1:GOTO 1220
 1190 Y=Y+1:GOTO 1220
 1200 X=X+1:GOTO 1220
 1210 X=X-1
 1220 IF X > 5 OR X < 1 OR Y > 5 OR Y < 1
      OR A > 5 OR A < 1 THEN PRINT "VOCE
      GANHOU":& T100,100:& T100,50:& T100
      ,50:& T75,66:& T100,66:& T75,66:& T
      60,255:GOTO 3000
 1230 HGR:HCOLOR=3:HPLOT0,0:CALL 62454:HC
      OLOR=0:GOSUB 120
 1240 FOR I=1 TO 6:IF MID$(S$(X,A),(Y-1)*
      6+I,1)="X" THEN NEXT:GOTO 1010
 1250 R=FC(FC,I)+1
 1260 HCOLOR=0:ON R GOSUB 125,130,135,140
      ,145,150
 1270 NEXT:GOTO 1010
 1280 INPUT "QUE DIRECAO 1-N 2-S 3-L 4-O"
      ;FC:IF FC <1 OR FC >4 THEN 1280
 1285 GOTO 1220
 1286 REM *              PERDIDO ?          *
 1290 INVERSE:HTAB 18:PRINT SX;"  ";SY;"
       ";SA:NORMAL:GOTO 1220
 1295 REM *      FIM POR DESISTENCIA        *
 1300 TEXT:HOME:END
 1400 REM *                                 *
 1410 REM *            INSTRUCOES           *
 1420 REM *                                 *
 2000 VTAB 12:HTAB 14:INVERSE:PRINT "PALA
      CIO DE CRISTAL":NORMAL:VTAB 22:INPU
      T "QUER INSTRUCOES ?";Y$: IF LEFT$(
      Y$,1) <> "S" THEN RETURN
 2010 HOME:VTAB 2:PRINT "O  OBJETIVO DO J
      OGO E' ACHAR A SAIDA DE":PRINT:PRIN
      T "UM PALACIO COM 5 ANDARES E 25 SA
      LAS POR":PRINT:PRINT "ANDAR, OU SEJ
      A,UM LABIRINTO ONDE APENAS"
 2020 PRINT:PRINT "UMA SALA CONTEM A SAID
      A, E  VOCE DEVERA":PRINT:PRINT "ENC
      ONTRA-LA NO MENOR TEMPO POSSIVEL.":
      PRINT:PRINT "OS COMANDOS SAO:"
 2030 PRINT:HTAB 6:INVERSE:PRINT "C";:NOR
      MAL:PRINT "-CIMA";:HTAB 17:INVERSE:
      PRINT "S";:NORMAL:PRINT "-SUL"
 2040 PRINT:HTAB 6:INVERSE:PRINT "B";:NOR
      MAL:PRINT "-BAIXO";:HTAB 17:INVERSE
      :PRINT "L";:NORMAL:PRINT "-LESTE"
 2050 PRINT:HTAB 6:INVERSE:PRINT "N";:NOR
      MAL:PRINT "-NORTE";:HTAB 17:INVERSE
      :PRINT "O";:NORMAL:PRINT "-OESTE"
 2060 PRINT:HTAB 6:INVERSE:PRINT "F";:NOR
      MAL:PRINT "-FIM";:HTAB 17:INVERSE:P
      RINT "V";:NORMAL:PRINT "-VISAO"
 2070 VTAB 23:INVERSE:PRINT "ESPACO";:NOR
      MAL:PRINT " PARA CONTINUAR"
 2080 IF PEEK(-16384) < 127 THEN 2080
 2090 POKE -16368,0:HOME:INVERSE:PRINT "V
      ";:NORMAL:PRINT " O COMPUTADOR PERG
      UNTARA EM QUE DIRE-":PRINT:PRINT "C
      AO VOCE DESEJA OLHAR."
 2100 PRINT:PRINT "POR FAVOR AGUARDE ENQU
      ANTO E GERADO  ":PRINT:PRINT "NOVO
      LABIRINTO":PRINT:PRINT:RETURN
 2500 REM *                                 *
 2510 REM *       MISSAO CUMPRIDA           *
 2520 REM *                                 *
 3000 TEXT:HOME:VTAB 5:HTAB 16:PRINT "PAR
      ABENS !!!"
 3010 PRINT:PRINT "VOCE CONCLUIU SUA MISS
      AO EM ";T;" SEGUNDOS"
 3030 VTAB 20:INPUT "QUER JOGAR NOVAMENTE
      ?";Y$
 3040 IF LEFT$(Y$,1) = "S" THEN RUN
 9990 NORMAL
 9999 DATA1,2,4,0,5,3,1,2,0,4,3,5,1,2,3,5
      ,4,0,1,2,5,3,0,4
```


APRENDENDO MATEMÁTICA

PARA LINHA APPLE

Este programa foi elaborado especialmente para crianças que estão se iniciando em aritmética. Trata-se de uma forma divertida de aprender as quatro operações básicas. A tela começa mostrando um menu para que o jogador escolha a operação desejada. Em seguida, o computador pergunta à criança o intervalo de valores permitido, a fim de propor problemas compatíveis com sua capacidade. Aí começa o jogo, com uma seqüência de operações, e só termina quando ocorrerem dez respostas corretas seguidas.

```
   10 REM ********************************
   20 REM *        PARA LINHA APPLE      *
   30 REM *                              *
   40 REM *    APRENDENDO MATEMATICA     *
   50 REM ********************************
   60 REM
  100 GOSUB 670 :   REM SUB-ROTINA DE SOM
  110 GOTO 260
  115 REM * GERA VALORES PARA A CARINHA *
  120 DIM X(100), Y(100)
  130 P=2*(355/113):FOR I=1 TO 100:ANGULO
      =P*(I/100):X(I)=15*SIN(ANGULO):Y(I)
      =15*COS(ANGULO):NEXT I
  140 RETURN
  150 POKE 230,32:CALL 62450:HGR:CALL -19
      94:GR:COLOR=7:PLOT 16,15:PLOT 24,15
      :COLOR=4:PLOT 20,19
  160 COLOR=11:IF C1=0 THEN 190
  170 PLOT 15,23:PLOT 25,23: PLOT 16,24:
      PLOT 24,24:PLOT 17,25: PLOT 23,25:
      HLIN 18,22 AT 26
  180 GOTO 200
  190 HLIN 18,22 AT 23:PLOT 17,24:PLOT 23
      ,24:PLOT 16,25:PLOT 24,25:PLOT 15,2
      6:PLOT 25,26
  200 COLOR = 1
  210 FOR I=1 TO 100:PLOT X(I)+20,Y(I)+20
      :NEXT I
  220 VTAB 22:HTAB 10:FLASH:IF C1=1 THEN
      PRINT " M U I T O  B E M !!":NORMAL
      :GOTO 250
  230 NORMAL:VTAB 22:HTAB 6:PRINT "DESCUL
      PE-ME, MAS ";B;" ";A$;" ";C;" = ";:
      INVERSE:PRINT E:NORMAL
  240 FOR I=1 TO 2000:NEXT I
  250 FOR I=1 TO 1500:NEXT I:HOME:HGR:POK
      E 34,0:HOME:TEXT:RETURN
  260 HOME:INVERSE:VTAB 10:HTAB 12:PRINT
      "APRENDENDO MATEMATICA":NORMAL:VTAB
      18:HTAB 4:PRINT "...AGUARDE"
  270 GOSUB 120
  280 HOME
  290 DIM OP$(4), SI$(4):FOR I=1 TO 4:REA
      D OP$(I):READ SI$(I):NEXT I
  300 FOR I=2 TO 8.STEP 2:VTAB I:HTAB 7:
      PRINT "PARA ";:INVERSE:PRINT OP$(I/
      2);:NORMAL:PRINT TAB(25) "DIGITE "
      SI$(I/2):NEXT I
  310 VTAB 12:HTAB 7:NORMAL:PRINT "SUA ES
      COLHA = ";
  320 GET A$:IF A$="" THEN 320
  330 IF A$<>"*" AND A$<>"+" AND A$<>"-"
      AND A$<>"/" THEN 320
  340 PRINT A$
  350 VTAB 17:HTAB 7:INPUT "MAIOR NUMERO
      = ";UL:VTAB 19:HTAB 7:INPUT "MENOR
      NUMERO = ";LL:R=UL+1-LL
  360 C=INT(RND(1)*R)+LL:B=INT(RND(1)*R)+
      LL
  370 IF A$="+" THEN DEF FN A(X)=B+C:GOTO
      430
  380 IF A$="-" THEN DEF FN A(X)=B-C:GOTO
      430
  390 IF A$="*" THEN DEF FN A(X)=B*C:GOTO
      430
  400 IF A$="/" AND C=0 THEN 360
  410 IF A$="/" AND INT(B/C) <> B/C THEN
      360
  420 IF A$="/" THEN DEF FN A(X)=B/C
  430 HOME:VTAB 7:HTAB 8:PRINT "RESPOSTAS
       CORRETAS EM SEGUIDA: ";:INVERSE:PR
      INT D:NORMAL
  440 E=FN A(X):VTAB 15:HTAB 15:PRINT B;"
       ";A$;" ";C;" = ";:INPUT F:IF F<> E
      THEN 480
  450 HOME:C1=1:GOSUB 150
  460 D=D+1:IF D=10 THEN 500:REM * FIM *
  470 GOTO 360
  480 HOME:C1=0:GOSUB 150
  490 D=0:GOTO 430
  500 REM * FIM DO JOGO (10 RESP.CERTAS)*
  510 B=0:GOSUB 560
  520 VTAB 22:HTAB 8:FLASH:PRINT "V O C E
       C O N S E G U I U ! "
  530 FOR I=1 TO 5:POKE 768,1:POKE 769,20
      0-I*30:CALL 770:NEXT I:FOR I=1 TO 1
      0:POKE 768,1:POKE 769,40+I*20:CALL
      770:NEXT I
  540 NORMAL:VTAB 24:HTAB 8:PRINT "QUER T
      ENTAR NOVAMENTE ? (S/N)";:GET A$:IF
      A$="S" THEN TEXT:RUN
  550 TEXT:HOME:HTAB 5:VTAB 8:PRINT "ATE
      LOGO !!!":END
```

```
560  POKE 230,32:CALL 62450:HGR:CALL -19
     94:GR
570  FOR J=1 TO 3
580  CL=0:L0=0:H1=19:S1=1:GOSUB 620
600  NEXT J
610  RETURN
620  FOR I=L0 TO H1 STEP S1:COLOR = INT(
     RND(1)*CL):X1=19-I:X2=20+I:Y1=19-I:
     Y2=20+I
630  HLIN X1,X2 AT Y1:VLIN Y1+1,Y2 AT X2
640  HLIN X2-1,X1 AT Y2:VLIN Y2-1,Y1 AT
     X1
650  NEXT I
660  RETURN
670  REM **      ROTINA DE MUSICA      **
680  FOR I=770 TO 795:READ M:POKE I,M:NE
     XT I
690  DATA172,001,003,174,001,003,169,004
     ,032,168,252,173,048,192,232,208,25
     3,136,208,239,206,000,003,208,231,0
     96
695  DATA"SOMAR","+","SUBTRAIR","-","MUL
     TIPLICAR","*","DIVIDIR","/"
700  RETURN
```

VOGAL OU CONSOANTE

PARA LINHA APPLE

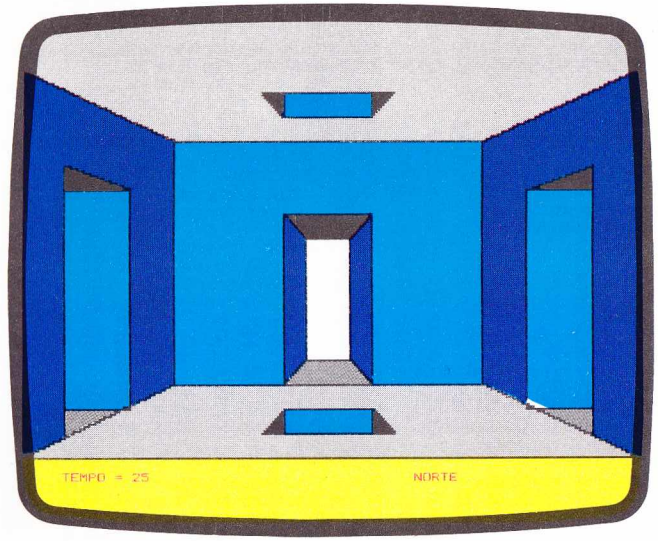
Este jogo, elaborado para dois ou mais participantes, vai empolgar toda a família. Disputado em grupo, revela-se alegre passatempo e bom exercício de vocabulário. Para cada participante, o computador fornece sete letras, cabendo ao jogador escolher o número de vogais e consoantes. Para tanto, antes de cada uma, deve digitar V ou C. Sua tarefa consiste, então, em formar com elas uma palavra no menor tempo possível e utilizando o máximo de letras. No final, os pontos obtidos aparecem na tela.

```
 10 REM****************************
 12 REM*     PARA LINHA APPLE     *
 14 REM*                          *
 16 REM*   VOGAL OU CONSOANTE     *
 18 REM****************************
 20 HOME
 25 GOSUB 2000
 30 INPUT "NUMERO DE JOGADORES: ";NP
 40 IF NP<1 THEN 20
 50 DIM PO(NP),C$(7),WO$(20)
 60 GOTO 1000
100 REM ** ESCOLHA DE LETRAS **
110 LE$=""
120 PRINT :PRINT "VEZ DO JOGADOR: "PL
130 PRINT :PRINT "V OU C ? F=FIM ":PRINT
140 FOR DU=1 TO 7
150 GET IN$:IF IN$="" THEN 150
160 IF IN$="F" THEN HOME :END
170 IF IN$<>"V" AND IN$<>"C" THEN 150
180 CH$="AEIOU":IF IN$="C" THEN CH$="B
    CDFGHJLMNPQRSTVXZ"
190 C$=MID$(CH$,INT(RND(1)*LEN(CH$)+1)
    ,1)
200 LE$=LE$+C$:PRINT C$;
210 NEXT DU
220 RETURN
300 REM ** ENTRADA DE LETRAS **
310 LE=0
320 PRINT :PRINT "TEMPO :"
330 FOR TI=45 TO 0 STEP -1
340 VTAB 3:HTAB 18:PRINT TI" "CHR$(7)
350 FOR DE=1 TO 55
360 IN=PEEK(-16384):IF IN<128 THEN 410
370 POKE -16368,0
380 IN$=CHR$(IN-128):VTAB 5:HTAB LE+1
385 IF IN$=CHR$(13) THEN RETURN
390 IF IN$>="A" AND IN$<="Z" THEN PRINT
    IN$;:LE=LE+1:WO$(LE)=IN$:GOTO 410
400 IF IN$=CHR$(8) AND LE>0 THEN LE=LE
    -1:HTAB LE+1:PRINT " ";
410 NEXT DE,TI
420 PRINT CHR$(7)+CHR$(7)+CHR$(7)
430 RETURN
500 REM ** CALCULA PONTOS  **
510 FOR C=1 TO 7
520 C$(C)=MID$(LE$,C,1)
530 NEXT
540 FOR CO=1 TO LE
550 FOR C=1 TO 7
560 IF WO$(CO)=C$(C) THEN C$(C)="":GOTO
    580
570 NEXT C:GOTO 600
580 NEXT CO
590 PO(PL)=PO(PL)+LE*TI
600 VTAB 7:HTAB 1
610 FOR P=1 TO NP
620 PRINT "JOGADOR "P": "PO(P)" PONTOS
    "
630 NEXT P
640 RETURN
1000 REM** PROGRAMA PRINCIPAL **
1010 FOR PL=1 TO NP
1020 GOSUB 100
1030 HOME :PRINT "FACA UMA PALAVRA A PA
     RTIR DISTO: "LE$
1040 GOSUB 300
1050 GOSUB 500
1060 NEXT PL
1070 GOTO 1010
1080 END
2000 REM ** INSTRUCOES **
2010 HOME :PRINT TAB(11)"VOGAL OU CONSO
     ANTE":PRINT :PRINT
2020 PRINT :PRINT "  O OBJETIVO DESTE J
     OGO E' CONSEGUIR"
2030 PRINT :PRINT "  CONSTRUIR PALAVRAS
      APARTIR DE LETRAS"
2040 PRINT :PRINT "  SORTEADAS PELO COM
     PUTADOR, TENDO O"
2050 PRINT :PRINT "  MAIOR NUMERO DE LE
     TRAS E O MAIS RA-"
2060 PRINT :PRINT "  PIDO POSSIVEL."
2070 VTAB 22:PRINT "  QUALQUER TECLA PA
     RA CONTINUAR ";
2080 GET A$:HOME
2090 RETURN
```


PALÁCIO DE CRISTAL

PARA LINHA APPLE

Sua missão neste intrincado jogo é escapar do palácio de cristal, onde todas as salas parecem iguais, mas apenas uma permite a saída. São cinco andares e 25 salas por andar, numa matriz 5 x 5 x 5. Você pode se movimentar em seis direções: para cima, tecle [C]; para baixo, [B]; norte, [N]; sul, [S]; leste, [L]; e oeste, [O]. Se estiver perdido, tecle [*] para obter sua posição no momento; e, em caso de desespero, a saída lhe será revelada por [?]. Para olhar em qualquer direção, pressione a [V] e indique a direção desejada. Quanto mais rápida for sua fuga, maior número de pontos você obterá.



Seis opções
No labirinto de salas iguais, com apenas uma saída, cada sala mostra cinco direções possíveis, embora sejam permitidas seis opções — uma delas encontra-se no plano da tela e, por isso, não aparece.

```
1    REM ********************************
2    REM *       PARA LINHA APPLE       *
3    REM *                              *
4    REM *     PALACIO DE  CRISTAL      *
5    REM ********************************
6    REM
7    REM *    INICIALIZA PALAVRA '&'    *
10   DATA201,084,208,015,032,177,000,032
       ,248,230,138,072,032,183,000,201
       ,044,240,003,076,201,222,032,177
       ,000,032,248,230
20   FOR I=768 TO 833:READ P:POKE I,P:NE
     XT I
30   DATA104,134,003,134,001,133,000,170
       ,160,001,132,002,173,048,192,136
       ,208,004,198
40   DATA001,240,007,202,208,246,166,000
       ,208,239,165,003,133,001,198,002
       ,208,241,096
50   POKE 1013,76:POKE 1014,0:POKE 1015,
     3
60   TEXT:HOME
70   GOSUB 2000
100  DIM FC(5,7):DIM FC$(5)
105  FC$(1)="NORTE":FC$(2)="SUL":FC$(3)=
     "LESTE":FC$(4)="OESTE"
110  FOR B=1 TO 4:FOR I=1 TO 6:READ FC(B
     ,I):NEXT:NEXT
115  GOTO 155
116  REM *                            *
117  REM * DESENHA SALAS DO LABIRINTO  *
118  REM *                            *
120  HPLOT 0,0 TO 279,0 TO 279,159 TO 0,
     159 TO 0,0 TO 69,29 TO 209,29 TO 20
     9,129 TO 69,129 TO 69,29:HPLOT 209,
     29 TO 279,0:HPLOT 209,129 TO 279,15
     9:HPLOT 69,129 TO 0,159:RETURN
125  RETURN
130  HPLOT 109,9 TO 169,9 TO 159,19 TO 1
     19,19 TO 109,9:HPLOT 119,19 TO 119,
     9:HPLOT 159,19 TO 159,9:RETURN
135  HPLOT 119,139 TO 159,139 TO 169,149
     TO 109,149 TO 119,139:HPLOT 119,139
```